

AD-A049 619

RCA CORP CAMDEN N J

F/G 17/2

STUDIES FOR DEVELOPMENT OF A UNIFIED NODE EMPLOYING DYNAMIC CHA--ETC(U)

DEC 77 B PATRUSKY, K BODZIOCH, R CHAN

F30602-75-C-0109

UNCLASSIFIED

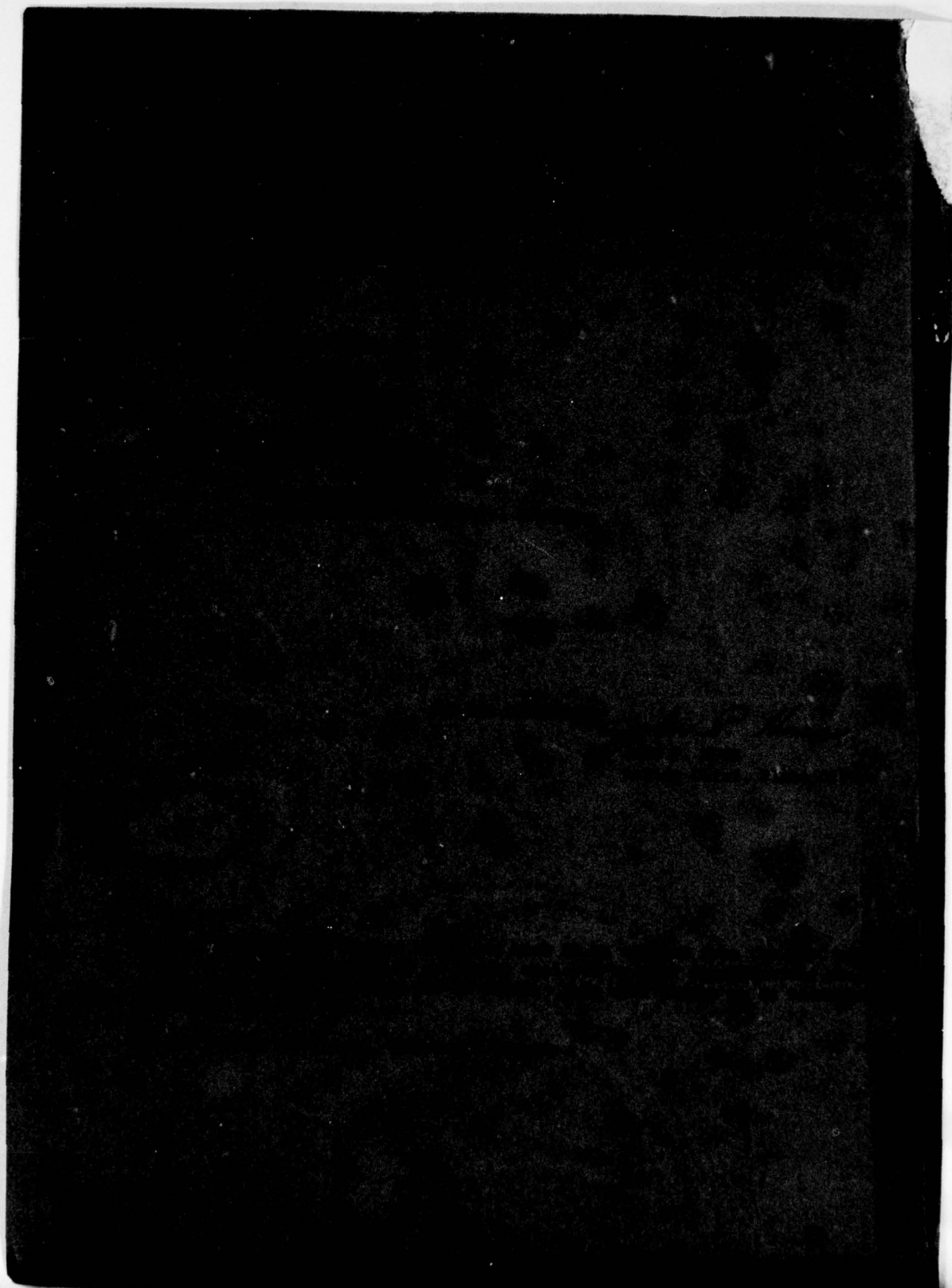
RADC-TR-77-380

NL

1 of 5
AD
A049619







UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-77-380	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) STUDIES FOR DEVELOPMENT OF A UNIFIED NODE EMPLOYING DYNAMIC CHANNEL ALLOCATION.		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report March 1975 - April 1977
7. AUTHOR(s) Bernard Patrusky Kenneth Bodzioch Robert Chan William Gesek Dan Heist		6. PERFORMING ORG. REPORT NUMBER N/A
8. PERFORMING ORGANIZATION NAME AND ADDRESS RCA Corporation Front & Cooper St. Camden NJ 08102		9. CONTRACT OR GRANT NUMBER(s) F20602-75-C-0109
10. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (DCLT) Griffiss AFB NY 13441		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 33126F 20220101
11. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Same		12. REPORT DATE December 1977
12. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		13. NUMBER OF PAGES 376
13. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) Same		14. SECURITY CLASS. (of this report) UNCLASSIFIED
14. SUPPLEMENTARY NOTES RADC Project Engineer: Frank Kozien (DCLT) (See Reverse)		15. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A
15. KEY WORDS (Continue on reverse side if necessary and identify by block number) Digital, Data Transmission, High Speed Digital, Multiple Access (Multi-subscrib- er), communications techniques, modulation and demodulation, communication operations, data processing.		
16. ABSTRACT (Continue on reverse side if necessary and identify by block number) RCA, during the period from March 3, 1975 until April 1977 has been engaged in studies, brassboard development, analysis and experiments whose objectives have been to prove the feasibility of the concept of a Unified Node Employing Dynamic Channel Allocation. The effort was guided by RADC statement of work PR C-5-2033.		

(Cont'd)

DD FORM 1473

1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

407380

1/p

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

20. (Cont'd)

The unified node concept is the concept of integrating circuit switch, message switch, and packet switch functions within a single node so that the node can provide appropriate switching and processing services to a wide variety of voice and data users.

T sub 1
The dynamic channel aspect of this concept deals with the composition of the internodal trunks. The internodal trunks employed are wideband trunk(s) (e.g. T_1 rates) which contain digitized voice time slots and non-real time data that is multiplexed in a frame structure and varied dynamically. Minimal restrictions are placed on channel data rates of voice data and the trunks are transparent to binary data transfers.

The key tasks in the contracted effort included:

- a) Studies of the unified node architecture, and dynamic channel allocation concepts leading to specification of a Unified Node Employing Dynamic Channel Allocation.
- b) Modification of the test facility developed under AF contract F30602-69-D-005C to support initial test and evaluation of the Dynamic Channel Allocation and Unified Node concepts and to prove that the dynamic channel concept proposed in the SOW was functionally/operationally feasible.
- c) Establishment of a simulator test facility having the flexibility to dynamically test a wide variety of nodal architecture configurations to evaluate and verify architectural performance and operational requirements.

18.

Qualified requesters may obtain copies of Appendices I through IV from RADC (DCLT) Griffiss AFB NY 13441.


Appendix I - HARDWARE DESIGN SPECIFICATIONS
Appendix II - SOFTWARE DESIGN HANDBOOK
Appendix III - EXPERIMENT RUN PRINTOUTS
Appendix IV - NODE ANALYTIC MODEL

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

EVALUATION

The work completed under Contract F30602-75-C-0109 has been evaluated and all program objectives have been met with highly satisfactory results. The significance of this effort is that it proved and validated the technical feasibility of an idea/concept for unified digital nodal switching architecture capable of handling digital voice, message and packet, bulk and narrative, traffic. The results derived from this effort directly relate to TPO #4A to the extent that its objective, development and validation of a technology base for use in the evolution of the future Defense Communications System (DCS) into a unified digital nodal configuration by the late 1980 time frame; have been met in a highly proficient technical manner. The results of this effort are intended for use and application to the local area, nodal area, and C³ area for the processing, control and distribution of digital voice, message and packet type communication traffic.


 FRANK KOZIEM
 Project Engineer

REVISION IN	
RTIS	White Section <input checked="" type="checkbox"/>
ODS	Boff Section <input type="checkbox"/>
QUANHOUSES	<input type="checkbox"/>
JUSTIFICATION	
BY	
DISTRIBUTION/AVAILABILITY CODES	
Dist.	AVAIL. CODE/SERIAL
A	

D D C
 RECEIVED
 FEB 3 1978
 RECEIVED
 D

TABLE OF CONTENTS

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
	Introduction	16
	Summary of Key Findings	20
1.0	Unified Node	25
1.1	Message Switching	26
1.2	Circuit Switching	27
1.3	Packet Switch Functions	28
1.3.1	Terminal Access Controller (TAC)	28
1.4	Node Central Control	29
1.4.1	Node Configuration Control	29
1.4.2	Route Control	29
1.4.3	Performance Assessment and Status Monitoring	30
1.4.4	Fault Diagnosis	30
1.4.5	Patch and Test	30
1.5	Dynamic Channel Allocation	30
1.6	Node Interfaces	31
1.7	Node Sizes	31
1.8	Traffic Characteristics	31
1.8.1	Traffic Rates	32
1.9	Packets vs Segments	32
2.0	Dynamic Channel Allocation	33
2.1	Frame Structure	33
2.1.1	Frame Structure Baseline	35
2.1.2	Frame and Slot Size	38
2.1.2.1	Voice Delay Considerations	40
2.1.2.2	Speed/Cost Considerations	42
2.1.2.3	Frame Rate/Slot Size Conclusions	42
2.1.3	Special Frame Boundaries	43
2.1.3.1	Class I Boundary	43
2.1.3.2	Class II Boundary	43
2.1.4	Initial Frame Pattern	46
2.1.5	Recommended Frame Structure	46
2.2	Bulk Data Handling	48
2.2.1	Specific Traffic Characteristics	48

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
2.2.2	Candidate Approaches	48
2.2.2.1	Approach 1 - Special Class III Service	49
2.2.2.2	Approach 2 - Handle Bulk as Class II Service	52
2.2.2.3	Approach 3 - Handle Bulk Data As A Version of Class I Traffic (See Figure 2.7)	54
2.2.3	Evaluation And Recommendation	59
2.2.4	Conclusions	61
2.3	Timing And Synchronization	63
2.3.1	Synchronization of The Basic Frame	63
2.3.1.1	Sync Acquisition	63
2.3.1.2	Sync Loss Detection	64
2.3.1.3	Controllable Variables	64
2.3.1.4	Bit Error Rate Environment	64
2.3.1.5	Equations of Probability	65
2.3.1.6	Definitions of Requirements And Constraints	65
2.3.1.7	Sync Recommendations	68
2.3.1.8	Use of Two Sync Patterns	72
2.3.1.9	Sync Pattern	72
2.3.2	Trunk Frame Synchronization	73
2.3.3	Maintaining Bit Integrity	79
2.3.3.1	Buffer Size	79
2.3.3.2	Clock Differential	81
2.3.3.3	Speed Buffering	81
2.3.3.4	Dynamic Channel Allocation	82
2.3.3.5	Buffer And Sync Requirements	82
2.3.3.6	Buffer And Channel Synchronization Recommendations	84
2.3.3.7	Timing Analysis	84
2.3.4	Independent Timing Sources	90
2.4	Impact of Dynamic Channel Allocation On System Parameters	92
2.4.1	Priority/Preemption	92
2.4.1.1	Priority of CCIS Traffic	93
2.4.2	Security	94
2.4.2.1	Class II Data Security	94
2.4.2.2	Class I Traffic Security	94

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
2.4.2.2.1	Link Encryption of Class I Traffic	95
2.4.2.2.2	Future Considerations	96
2.4.3	Routing	96
2.4.4	Call Setup And Breakdown	97
2.4.4.1	Call Setup	97
2.4.4.2	Call Breakdown	99
2.4.4.3	Preemption	99
2.4.4.3.1	Destination Call Preemption	99
2.4.4.3.2	Trunk Channel Preemption	100
2.4.5	Service Features	101
2.4.5.1	Class I Service Features	101
2.4.5.1.1	Attendant/Recall Operator Function	101
2.4.5.1.2	Conferencing	102
2.4.5.1.3	Precedence/Preemption	103
2.4.5.2	Class II Service Features	103
2.4.5.2.1	Accountability	104
2.4.5.2.2	Multiple Addressing	105
2.4.5.2.3	Transaction Preemption	105
2.4.5.2.4	Packet Circuit Type	106
2.4.6	Technical Control	107
2.4.7	Network Management	108
2.5	Useful Range of Operation	108
2.5.1	Maximum Trunk Rate	109
2.5.2	User Channel Rate	110
3.0	Common Channel Interoffice Signalling (CCIS)	113
3.1	CCIS Approach Studies	113
3.1.1	CCIS Transmission Approaches	113
3.1.1.1	In Band Approach	113
3.1.1.2	Dedicated Out of Band Approach	114
3.1.1.3	Use of Asynchronous CCIS Packets	114
3.1.2	CCIS Packet Blocking and Delay Problems	115
3.1.3	Frame Changing	116
3.1.3.1	Protection Against Non-Coordinated Frame Lists	117
3.1.3.2	Conflict Resolution	118

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
3.1.4	Concatenation of CCIS Messages	119
3.1.5	Recommended Approach Summary	120
3.2	Message Formats	121
3.2.1	Class I Message Formats	122
3.2.1.1	Modification To Basic TTC-39 Format	123
3.2.1.2	Additional Potential Modification	123
3.2.1.3	Future Modification	129
3.2.2	Control Messages For Class II Traffic	129
3.2.2.1	Transaction Descriptions	132
3.2.2.1.1	Call Request (Similar To Call Initiate)	132
3.2.2.1.2	Call In Progress Packet	132
3.2.2.1.3	Call Terminate	132
3.2.2.1.4	Single Packet Call	132
3.2.2.1.5	Packet Acknowledgement	132
3.2.2.1.6	Request For Next Message	132
3.2.2.1.7	Buffers Full	133
3.2.2.1.8	Tables Full	133
3.2.2.1.9	Called Line Unavailable	133
3.2.2.1.10	Called Line Is Out of Service	133
3.2.2.1.11	Invalid Address	133
3.2.2.1.12	Invalid Security	133
3.2.2.1.13	Invalid Sequence Number	133
3.2.2.2	Transaction Descriptions	134
3.2.2.2.1	Data Packet Header Format	134
3.2.2.2.2	Packet Response Format	139
3.2.2.3	Rationale For Recommendations	141
3.2.2.3.1	BCD Addressing	142
3.2.2.3.2	Format Identifier	142A
3.2.2.3.3	Extraneous Overhead	143
3.2.2.3.3.1	Net Addresses	143
3.2.2.3.3.2	Route Control	143
3.2.2.3.3.3	The Node Control	143
3.2.2.3.3.4	Route Tabulation	144
3.2.2.3.4	Redundant Data	144

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
3.2.2.3.5	Format Features Which Aid Packet Processing	144
3.2.2.3.6	Format Features Which Aid Packet Protection	145
3.2.2.3.7	Reason For Compressed Response Packet	145
3.2.2.4	Possible Use of Singular Transaction Identifiers	146
3.2.2.5	Addendum	146
4.0	Interfacing The ADPT Testbed To Existing Networks	149
4.1	ADPT Operation As A Host of The ARPA Network	150
4.1.1	Host-Host Protocol	154
4.1.1.1	Host-To-Host Design Philosophy	154
4.1.2	Telecommunication Network (TELNET) Protocol	156
4.1.3	Network Control Program (NCP)	159
4.1.3.1	NCP System Calls	160
4.1.3.2	Control Commands	163
4.1.4	The Logging - In Process	164
4.1.4.1	Logging - In Exchange	164
4.1.5	User Identification	167
4.1.6	ARPA Net Leader Format	168
4.1.7	IMP Header Format	168
4.1.8	Local/Distant/Very Distant Host Interfaces	
	Local Host	168
4.1.9	Special Provisions For Very Distant Host	172
4.1.9.1	Reliable Transmission Package	174
4.1.9.2	Error Detecting Special Host Interface Packet	
	Format On Line	176
4.1.9.2.1	Cyclic Redundancy Check	180
4.1.10	Refer To Appendices A and B	180
4.2	AUTODIN Interface	182
4.2.1	Pseudo Host	184
4.2.1.1	Pseudo Host Processing Input (From Message Switch Software)	184
4.2.1.2	Pseudo Host Processing - Output	185
4.2.1.3	Interface to Message Switch	186
4.2.1.4	Terminal Routing	189
4.2.2	Message Switch Program	189

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
4.2.2.1	Executive Program	192
4.2.2.2	Message Linkage From Input Line to Output Line	193
4.2.2.3	Line Management - Processing Program Interface	196
4.2.2.4	Input Processing	197
4.2.2.5	Output Processing	199
4.2.2.6	Message Exchange	201
Appendix A	Arpanet Host - IMP Leader Format	204
Appendix B	Host Interface Leader/Header Formats For ADPT Testbed Simulator	217
5.0	ADPT Testbed and Experiment Descriptions	224
5.1	Processing Components	226
5.1.1	Communications Processor	226
5.1.2	Memory Structure	226
5.1.2.1	Main Memory Structure	226
5.1.2.2	Read-Only Memory	226
5.1.2.3	Scratchpad Memory	226
5.1.3	Communications Channel	226
5.1.3.1	Bootstrap	226
5.1.3.2	Operator's Console	226
5.1.4	Input/Output Processor (IOP)	226
5.1.4.1	Peripheral Devices	227
5.2	Specialized Switching Components	227
5.2.1	Time Division Multiplex (TDM) Matrix	227
5.2.2	Interactive Communications Channel (ICC)	229
5.2.2.1	ICC Interfaces	229A
5.2.3	Buffer Matrix (BMX)	231
5.2.3.1	BMX Interfaces	233
5.2.4	Character Buffer Subsystem (CBSS)	233
5.3	Three Node Subnetwork Simulator	234
5.3.1	Integrated Switching	234

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
5.3.1.1	Dynamic Channel Allocation	236
5.3.1.2	Common Channel Interoffice Signalling (CCIS)	238
5.3.1.3	Narrative/Packet Interchange	238
5.3.2	Circuit Switch	239
5.3.2.1	Local Calls	239

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
5.3.2.2	Remote Calls	239
5.3.2.3	Tandem Calls	240
5.3.3	Message Switch	241
5.3.3.1	Protocols And Formats	241
5.3.3.2	Priority Handling	241
5.3.3.3	Security	242
5.3.3.4	Routing	242
5.3.4	Packet Switch	242
5.3.4.1	Protocols And Formats	242
5.3.4.2	Priority Handling	243
5.3.4.3	Security	243
5.3.4.4	Routing	244
5.4	Traffic Sources And Sinks	244
5.4.1	CVSD Phones	244
5.4.2	Silent 700 Terminals	244
5.4.3	Host Simulators	245
5.5	Test Node Simulator	245
5.5.1	Test Node Simulator Software Package	245
5.5.1.1	Microscopic Simulation	246
5.5.1.2	Macroscopic Simulation	246
5.5.2	Traffic Generation	246
5.5.3	Network Simulation Software	249
5.5.3.1	Simulator Control Unit (SCU)	251
5.5.3.2	Traffic Generator Unit (TGU)	251
5.5.3.3	Data Entry/Data Sink (DEDS)	253
5.5.3.4	ICC Simulator Unit (ISU)	253
5.5.3.5	Traffic Record Unit (TRU)	254
5.6	Experiment Variables	254
6.0	Analysis and Evaluation of Experimental Results	256
6.1	Experimental Model Description	256
6.1.1	Node Simulator Configuration	257
6.1.2	Node Simulator Operation	257
6.2	Results	260
6.3	Delays Encountered in Node	262
6.3.1	Processing Delays	262

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
6.3.2	Trunk Queue Delays	266
6.3.3	Transmission Delays	269
6.3.4	Impact of Priority Handling	271
6.4	Buffer Utilization	273
6.4.1	Predetermined Threshold Throttling	273
6.4.2	Natural Throttling	273
6.5	Trunk Utilization	274
6.5.1	Voice	274A
6.5.2	Combined Trunk Utilization	275
6.6	Thruput	275
6.7	Conclusions	275
7.0	Future Node Architecture - Hardware	279
7.1	Introduction	279
7.2	Node Requirements	281
7.2.1	Node Traffic	281
7.2.2	Switch Functions	281
7.2.3	Node Interfaces	282
7.2.3.1	Voice Subscriber Lines	282
7.2.3.2	Synchronous Data Sources	283
7.2.3.3	Asynchronous Data Sources	283
7.2.4	Internode Trunks	283
7.2.5	Node Size	283
7.2.6	Performance Requirements	286
7.3	Functional Overview	286
7.3.1	Message Switch Functions	290
7.3.1.1	Special Autodin Terminal Interface	291
7.3.2	Circuit Switch	291
7.3.2.1	Gateway Functions	291
7.3.2.2	Access Interface and Matrix Functions	292
7.3.2.3	Controllers, Scanner, Tone Generators and Receiver Senders	293
7.3.3	Packet Switch	294
7.3.3.1	Link Access Protocol	296
7.3.3.2	Segment Validation	297
7.3.3.3	Segment/Packet Format Conversion	297

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
7.3.3.4	Source Node Call Control	298
7.3.3.5	Packet Route	298
7.3.3.6	Packet Validation	298
7.3.3.7	Destination Node Call Control	298
7.3.3.8	Packet/Segment Conversion	299
7.3.3.9	Segment Transmit	299
7.3.3.10	Packet Transmit	299
7.3.3.11	CCIS Message Processing	299
7.3.3.12	Summary	299
7.4	Processor Candidates	300
7.4.1	Single Processor	300
7.4.1.1	Modified Single Processor	302
7.4.1.1.1	Interactive Communications Channel	304
7.4.1.1.2	Buffer Matirx	306
7.4.2	Multiple Processor	309
7.4.2.1	Multiple Processors without Shared Memory	309
7.4.2.2	Multiple Processors with Shared Memory	314
7.4.3	Multiprocessor	316
7.4.3.1	Multibus Multiprocessor	320
7.4.3.2	Multiprocessor with Multiport Memory and and Central BMX	324
7.4.3.3	Multiprocessor with Multiport Memory and Decentralized BMX	324
7.4.3.4	Multibuss/Multiport Shared Memory	328
7.4.3.4.1	Multibuss/Multiport Shared Memory with Independent BMX	330
7.4.3.4.2	Multiport/Multibus Shared Memory with Circuit Switch	330
7.5	Evaluation	335
7.5.1	Evaluation Criteria	335
7.5.2	Criteria Ranking	336
7.5.3	Candidate Evaluation	336
7.6	Recommended Architecture	340
7.6.1	Processing Requirement	340
7.6.2	BMX Requirement	340
7.6.3	Voice Shared Memory Requirements	341

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
7.6.4	ICC Requirements	342
7.6.5	Shared Data Memory Size and Speed	344
7.6.6	Processor Characteristics	345
7.6.6.1	Memory Addressing Capability	346
7.6.6.2	DMA Capability	347
7.6.6.3	Asynchronous Memory Operation	347
7.6.6.4	External Interrupts	347
7.6.6.5	Software Support Features	348
8.0	Future Node Architecture - Software	349
8.1	General	349
8.2	Executive Structure	351
8.2.1	Master-Slave Organization	351
8.2.2	Autonomous Organization with Multiple Executive	351
8.2.3	Organization with Floating Executives	352
8.2.4	Executive Structure for Packet Switch	352
8.3	Interrupt Structure	354
8.3.1	No Program Interrupts	354
8.3.2	Program Interrupt to all Processors	355
8.3.3	Program Interrupt to One Processor	355
8.3.4	Interrupt Structure for Packet Switch	356
8.4	Processing Allocation	356
8.4.1	Packet Switching	357
8.4.1.1	Interrupt Servicing Level	357
8.4.1.2	Interrupt Processing Level	358
8.4.1.2.1	Process Transmitted Block	358
8.4.1.2.2	Process Received Block	358
8.4.1.3	Base Processing Level	360
8.4.1.3.1	Access Lines	361
8.4.1.3.2	Output Processing	362
8.4.1.3.3	Monitor	362
8.4.2	Voice Switching	362
8.4.2.1	Access Line Processing	362
8.4.2.2	Trunk Processing	363
8.4.2.3	Control of BMX	363

<u>PARAGRAPH</u>	<u>TITLE</u>	<u>PAGE #</u>
9.0	Recommendation for Future Studies/Development	365
9.1	Replacement of Lab Processor	365
9.2	Enhancement of Hosts	365
9.3	Addition of A COMSEC Subsystem	365
9.4	Gateways	366
9.5	Satellite Interface	366
9.6	Requirements	366
9.7	Further Experimental Tests	366
9.8	Routing Algorithm	367
9.9	Addition of Terminal to Testbed	367
9.10	Network Studies	367
9.11	Test Subnetwork	367
9.12	Standalone BMX	367
9.13	Multiprocessor Configuration	367
9.14	Sync/Frame Change Procedure	367
9.15	Voice/Data Priorities	368
10.0	Conclusions	368
Glossary		369
Appendix I	Hardware Design Specifications	
Appendix II	Software Design Handbook	
Appendix III	Experiment Run Printouts	
Appendix IV	Node Analytic Model	

LIST OF FIGURES

<u>FIGURE</u>	<u>TITLE</u>	<u>PAGE #</u>
2.1	Multiplex Structure of Internodal Trunks	34
2.2	Frame Without Class II Traffic	37
2.3	Packet Splitting	39
2.4	End-to-End Class I Traffic Delays	41
2.5	Frames With Class II Limit	45
2.5A	Recommended Frame Structure	47
2.6	Frame Containing Class III Data Packets	53
2.7	Frame Containing Class III Synchronous Calls	55
2.8	Trunk Frame List	74
2.9	Frame Change Sequence	76
2.10	Buffer Arrangement Multinode Path	80
2.11	Channel Movement Within Frame	83
2.12	Timing and Buffer Requirements Incoming Trunk/Outgoing Loop	85
2.13	Timing and Buffer Requirements Outgoing Trunk/Incoming Loop	86
2.14	Timing and Buffer Requirements Tandem Call	87
3.1	Message Format for Call Initiate	124
3.2	Message Format for Call Complete	125
3.3	Message Format for Call Answer, Call Release, Etc.	126
3.4	Message Format for Acknowledgement, Neg-Acknowledge Etc.	126
3.5	Message Format for Invalid Route	127
3.6	Message Format for Sync A and Sync B	127
3.7	Message Format for Interface Complete	128
3.8	Binary Segment Leader	148
4.1	ADPT Host Interfaces to ARPA Network	151
4.2	Host to Host Connection Between Processor	155
4.3	TELNET Dialog	158
4.4	NCP and TELNET Network Control	161
4.5	Socket Number Elements	166
4.6	ARPA Net Leader Format	169
4.7	IMP Header Format	171
4.8	Implementation of Very Distant Host to IMP Interface	173

<u>FIGURE</u>	<u>TITLE</u>	<u>PAGE #</u>
4.9	Control Word Format for use on very Distant Host Communication Line	175
4.10	Segmentation of a Message Into Packets	177
4.11	Packet Format on Distant Host Line	178
4.12	Message/Package Interface	183
4.13	Logical Configuration of Message Switch in Testbed	188
4.14	JANAP 128 Header	190
5.1	Three Node Subnetwork Configuration (Physical Configuration)	225
5.2	Interactive Communications Channel Interfaces	230
5.3	Deleted	
5.4	Three Node Subnetwork Function Configuration	235
5.5	Multiplexed Frame of an Internodal Trunk	237
5.6	Testbed Applications	247
5.7	Addition of Network Simulation Software	248
5.8	Test Simulator Software	250
5.9	Experiment Software Processing and Timing	252
6.1	Node Simulator Configuration	258
6.2	Nodal Time in Minutes Versus Voice Call in Progress	263
6.3	Nodal Time in Minutes Versus Data Transactions In Progress	264
7.1	Unified Node Functional Block Diagram	289
7.2	Packet Switch Functions	295
7.3	Single Processor .	301
7.4	Modified Single Processor	303
7.5	ICC Block Diagram	305
7.6	BMX Block Diagram	307
7.7	Basic Multiple Processor Configuration	310
7.8	Interprocessor Communication Links	311
7.9	Multiple Processor with High Speed Buss	313
7.10	DEC PDP-11 Configuration	315
7.11	Varian Configuration	317
7.12	Data General Eclipse Configuration	318

<u>FIGURE</u>	<u>TITLE</u>	<u>PAGE #</u>
7.13	Basic Multiprocessor	319
7.14	Multibus Multiprocessor	321
7.15	Pluribus System Configuration	322
7.16	Multiprocessor with Multiport Memory and Central BMX	325
7.17	Multiprocessor with Multiport Memory and Decentralized BMX	327
7.18	Multibuss/Multiport Shared Memory	329
7.19	Interdata Multiprocessor Configuration	331
7.20	Multibuss/Multiport Shared Memory with Independent BMX	332
7.21	Multibuss/Multiport Shared Memory with Circuit Switch	333

LIST OF TABLES

<u>TABLE</u>	<u>TITLE</u>	<u>PAGE #</u>
2.1	Bulk Data Traffic Characteristics	51
2.2	Bulk Data Handling Approach Comparisons	60
2.3	Formula for Sync Acquisition	66,67
2.4	Calculated Probabilities	69,70
3.1	Packet Data Call Header	130
3.2	Packet Response	131
4.1	Message Types in ARPA Network	170
6.1	Node Simulator Performance Record	259
6.2	Experimental Results	261
6.3	Computation of Queue Delays	268
6.4	Queue Wait Times	270
6.5	Computation of Processing Delays	272
7.1	Node Sizes	285
7.2	Autodin II Acceptance Delay	287
7.3	Autodin II Delivery Delay	288
7.4	Intercommunication Link Trade Matrix	312
7.5	Architecture Trade Matrix	337

INTRODUCTION

RCA, during the period from March 3, 1975 until April 1977 has been engaged in studies, brassboard development, analysis and experiments whose objectives have been to prove the feasibility of the concept of a Unified Node employing Dynamic Channel Allocation. The effort was guided by RADC statement of work PR NO. C-5-2033.

The unified node concept is the concept of integrating circuit switch, message switch, and packet switch functions within a single node so that the node can provide appropriate switching and processing services to a wide variety of voice and data users.

The dynamic channel aspect of this concept deals with the composition of the internodal trunks. The internodal trunks employed are wideband trunk(s) (e.g. T-1 rates) which contain digitized voice time slots and non-real time data that is multiplexed in a frame structure and varied dynamically. Minimal restrictions are placed on channel data rates of voice data and the trunks are transparent to binary data transfers.

The key tasks in the contracted effort included:

- a) Studies of the unified node architecture, and dynamic channel allocation concepts leading to specification of a Unified node employing Dynamic Channel Allocation.
- b) Modification of the test facility developed under AF contract F30602-69-D-005C to support initial test and evaluation of the Dynamic Channel Allocation and Unified Node concepts and to prove that the dynamic channel concept proposed in the SOW is functionally/operationally feasible.
- c) Establishment of a simulator test facility having the flexibility to dynamically test a wide variety of nodal architecture configurations to evaluate and verify architectural performance and operational requirements.

These key tasks have been completed. Results of this effort are presented in this final report and in the developed specification for a future unified node. The residual brass-board testbed facility has been configured to demonstrate a barebones unified node subnetwork offering message switching, packet switching, and circuit switching to users and employing Dynamic Channel Allocation on internodal trunks.

Additionally the testbed has been reconfigured to include a Network simulator software package to provide for testing of the developed test node. This experiment provides data as to delays, thruput, buffer utilization, etc. experienced by the test node under different incident voice and data traffic loads and different voice blocking probabilities and buffer throttling conditions. Data acceptance and delivery delays are measured as a function of precedence category.

This final report is organized in the following manner.

Section 1.0 is a brief description of the unified node functions and requirements. We will then discuss how we examine specific study tasks.

Section 2.0 addresses a series of contracted study tasks regarding dynamic channel allocation leading to such recommendations as the frame rate and structure of the trunk frame, a method for dynamically changing frames, and a proposed approach for maintaining bit integrity (timing and synchronization).

Section 3 addresses the problem of Common Channel Interoffice Signalling (CCIS) signalling and proposes an integrated approach using a combination of packet and out of band trunk signalling to effect intertrunk voice connections. Formats for Class I CCIS packets and Class II control and data packets are also evaluated leading to proposed formats.

Section 4 proposes methods for interfacing the testbed simulator with the ARPA and Autodin networks. It is noted that the Autodin interface was incorporated in the testbed, but the ARPA interface was not.

Section 5 describes the testbed from a physical and functional viewpoint describing both the three node subnetwork configuration used to demonstrate the node with dynamic channel allocation and the experiment setup used to obtain data to evaluate the concept.

Section 6 analyzes the delays and other performance parameters of the dynamic channel allocation concept based on the results of the analytic model developed and presented as Appendix IV and the experiment results (Appendix III) obtained from the test facility.

Section 7 and 8 trades various hardware and software architectures and develops a recommended configuration for the future unified node.

Section 9 lists and briefly discusses recommendations for further studies, tests, hardware development, etc.

Section 10 lists conclusions.

Appendix I contains hardware specifications placing emphasis on the Interactive Control Channel (ICC) and Buffer Matrix (BMX) units developed specifically for this phase of the program.

Appendix II is the software design handbook employed to develop the software code for the testbed data switch programs. This handbook does not address software packages in the testbed which were developed in previous phases of the contract. Key software packages developed included:

- a) Advanced Data Communication Control Procedure (ADCCP) protocol handler
- b) Segment Interface Protocol Handler
- c) AUTODIN psuedo host gateway
- d) Network Simulator (for experimental tests)

Appendix III - Includes computer printouts of experiment data. It is noted that the versatile simulator tool developed can yield a great deal more data on the impact of dynamic channel allocation for different nodal architectures than time and cost constraints permitted. This is a particularly fruitful area for follow on investigations.

Appendix IV is a queuing model developed by personnel in RCA Princeton labs which was effectively utilized to provide insight into the impact of the dynamic channel allocation concept on trunk queue waiting time and serves to validate the experimental model.

RCA is pleased to submit this report which we feel represents a substantial step forward in an area of investigation of high challenge, technical interest, and future potential.

SUMMARY OF KEY FINDINGS

HIGHLIGHT

TESTBED DEMONSTRATION

A barebone unified subnetwork employing Dynamic Channel Allocation on trunks has been successfully demonstrate by the ADPT testbed, The Testbed provides for:

- 2 Hosts (sources of segment traffic)
- 2 Mode II terminals (source of message traffic)
- 4 Continuous Variable Slope Delta Mod (CVSD's)
(source of digital voice)

Hosts and mode II terminals can converse over internodal trunks with voice traffic simultaneously multiplexed on the trunks. Hosts and mode II terminals can communicate both with each other and from terminal to host and host to terminal (see section 5.0).

SPECIAL FRONT END HARDWARE

Two devices developed by RCA for this program have proven their value in any future development of a unified node.

The Interactive Communication Controller (ICC) loads segments or packets to and from data memory via the direct memory access technique and performs all bit processing and error check functions prescribed by the ADCCP protocol, thus off loading the main processor (see Section 5 - Testbed Description, and also the ICC specification - see Appendix I). Each ICC can interface up to 128 host or data trunk lines, and support a 2 MBPS thru-put.

The Buffer Matrix (BMX) accepts digital voice and data trunk inputs to interface internodal trunks. The BMS dynamically accepts frame lists from a processor and proceeds independently to service all channels and formats the designated trunk frame (see Section 5 - Testbed Description, and also the BMX specification - Appendix I). Each BMX can handle up to 128 voice or data channels and supports an 8 MBPS thru-put.

These devices are necessary since they handle repetitive high speed processing functions best handled by ROM sequencers or special firmware and which would present an awesome processing burden if performed by the main processing system.

TRUNK FRAME STRUCTURE

The unified node concept employing dynamic channel allocation is feasible. A trunk operating at a T-1 carrier employing a 15,440 bit frame repeated at 100 frames per second is a good compromise between allowable voice traffic delays and practical speed limitations (see Section 2.1).

CCIS SIGNALLING AND SUPERVISION

CCIS packets are the best means for handling voice signalling and supervision. These packets should employ formats wherein existing Out of Band (OOB) signalling and supervision standard message formats form a subset of the CCIS packet formats (see 3.1, 3.2).

SYNC/FRAME CHANGE PATTERN

A dual sync (sync A/sync B) leading each trunk frame provides an efficient means for synchronizing frames and coordinating frame changes (switching from frame list A to B and vice versa) (see 2.3). A 32 bit dual sync pattern can be employed and meet reasonable sync acquisition and dropout criteria (see 2.3).

BULK DATA HANDLING

Bulk data should be handled as class II low priority data (see 2.2).

IMPACT OF DYNAMIC CHANNEL ALLOCATION OF SYSTEM PARAMETERS

Dynamic Channel Allocation introduces certain special considerations as regards to call establishment/disestablishment, data misrouting, routing algorithms, and user service features. (See 2.4).

SPEED LIMITATION

A 15 MBS trunk would be a practical limit for this concept based on present day memory speeds, however, above T-1 normal super-group multiplexing is more practical (see 2.5).

FRAME BOUNDARIES AND LIMITS

The class I/II boundary should not require a physical marker within the frame. A class II programmable limit should be introduced to prevent unusual data surges on trunks. A class I programmable limit is required to adjust voice grade of service (see 2.1).

VOICE BUFFER REQUIREMENTS

In order to maintain bit integrity approximately two frame channel slots of buffering are required to buffer local to trunk and trunk to local digital voice traffic and three channel slots for tandem traffic.

FRAME CHANGE COMMAND

A concatenated frame change command is recommended to permit simultaneous multiple changes to the frame to speed the frame change process. A master/slave approach is recommended to eliminate the glare problem (both ends of link attempting to simultaneously change frames) and to provide assurance that frames remain coordinated at both link ends (see 3.1).

CCIS PRECEDENCE

CCIS packets should be marked as highest precedence packets to minimize call connect/breakdown times (see 3.1).

POINT OF COMPLETING CALL CONNECTION

Reserved voice bandwidth can be utilized for class II data transfers until the call is answered (see 2.4).

TRANSMISSION OVERHEAD

Transmission overhead on trunks is largely a function of packet size running at about 17% for packets if maximum size is 2000 bits or 8.5% if maximum packet size is 4000 bits (see 2.2).

A key feature of the study is the recommendation to permit packaging splitting so that the trunk frame can be fully utilized without added overhead penalty anytime sufficient data is backlogged to fully occupy the frame (see 2.1).

VOICE DATA RATE CONSTRAINTS

It is recommended that class I traffic be handled in 8 bit groups. Coupled with a frame rate of 100 frames per second any multiple of 800 bps up to the maximum limit would be theoretically acceptable (see 2.1).

User rates must be constrained to much less than the full trunk rate. For a T-1 trunk, a 200 KBS limit is the recommended maximum user rate (see 2.5).

POTENTIAL BANDWIDTH SAVINGS DUE TO DYNAMIC CHANNEL ALLOCATION

Dynamic Channel Allocation can save that bandwidth normally required to handle normal Bulk 1 and narrative data traffic, however, the small end to end delays required for control messages, flash and flash override traffic, and interactive traffic requires that dedicated data channels be reserved on the internodal trunk for this traffic (which may constitute from 10-50% of the busy hour data traffic volume). For this reason three priorities of traffic are suggested for data transmission.

- Priority 1) All control and messages and flash override traffic
- Priority 2) Interactive Traffic (other than flash)
- Priority 3) Narrative and Bulk traffic (see 6.)

NODE ARCHITECTURE

A multiprocessor shared memory configuration is recommended wherein data traffic and voice traffic employ separate shared memories (see 7.0).

PROCESSING LOAD DISTRIBUTION

Load Distribution among processors should be based on assignments of processors to handle data of specific lines and trunks. However, specific line/trunk assignments should be dynamically controlled (see section 8).

EXECUTIVE CONTROL

A single master executive controller is recommended to supervise node load distribution and initialization and recovery functions. The executive control function should be taken over by backup processors automatically in the event of failure (see section 8).

1.0 UNIFIED NODE

This section provides a brief overview of the unified node and discusses some of the basic node requirements. The unified node provides for message switching, circuit switching, and packet switching. These services are designed to accommodate a wide range of voice and data users. These unified nodes are to be modular in size and are intended to serve as backbone nodes of the future DCS, and also as base communication centers for military bases.

The unified node will be a versatile, modular communications switching node which will:

- a. Provide for secure and non-secure circuit switching of Class I traffic
- b. Accommodate a wide variety of Class I traffic, such as variable rate digital voice subscribers, high speed FAX, slow scan TV, and interactive graphics.
- c. Provide for message store and forward switching to handle narrative record traffic.
- d. Provide for packet switching to handle asynchronous but near real time data transactions to accommodate the following needs:
 - Man/Man (Terminal/Terminal) communications including both interactive type communications and standard narrative/record communications.
 - Man/Computer (Terminal/Computer) query response communications.
 - Computer/Computer communications ranging from transfer of short interactive traffic to transfer of very lengthy transaction (bulk).
 - CCIS signalling for establishment and disestablishment of Class I calls.
 - Class II control messages to control packet and/or

message accountability, flow control, configuration control, transfer of statistics, etc.

- e. Provide for Dynamic Channel Allocation. A key feature of the node is to provide for a single type wideband trunk which can contain all classes and types of traffic multiplexed in a highly efficient fashion. These trunks will be the sole means for internodal communications and offers the following advantages:

- Use of few wideband trunks as opposed to many narrowband trunks should offer significant advantages in terms of rental costs, and cost for design and maintenance of the tech control subsystem.
- Many different data rates of Class I traffic may have to be accommodated by the node. Ability to compact traffic on internodal trunks by providing for proportional size channel slots in the multiplexed frame permits efficient utilization of the transmission facility.
- A reduction in transmission efficiency and a significant improvement in non-peak hour performance is achievable by permitting data to be transmitted over unoccupied voice channels.

1.1 MESSAGE SWITCHING

The message switch function may not be required at all network nodes, therefore, a modular approach toward incorporating this function is desired. The message handling and storage functions required will not differ in basic concept or size from Autodin I Message Switching Centers. The key differences shall involve the interfacing to terminals and other message switches. Rather than interfacing directly with terminals and switches, the Unified Node Message switch will interface subscribers indirectly, via the packet switch function in the node. Another

factor shall be the integration of the message switch supervisory functions with the supervision functions of the circuit and packet switch functions.

1.2 CIRCUIT SWITCHING

The circuit switch will be processor controlled. This switch will differ from the conventional circuit switch in several major areas.

- All traffic from subscribers will be digital and signalling and supervision will be digital. Interfaces to other analog networks will be provided on a limited basis as gateway interfaces to the basic switch.
- CCIS signalling will be accomplished employing signalling packets which are generated and detected by the circuit switch processing function and transmitted to other nodes in the network via the packet switch functions.
- Since all traffic is digital and the switch must be able to handle a wide range of digital user rates, the circuit switch will service many types of Class I traffic (secure voice, non-secure voice, dialed data, fax, interactive graphics, slow scan video, etc.).
- Since relatively slow speed subscribers of different data rates are to be multiplexed onto the wideband internodal trunk a buffered memory switching approach will be employed to provide for loop to trunk speed conversion and to delay subscriber data access to the trunk until the channel allocated on the trunk for that subscriber is ready to transmit.
- The circuit switch processing function will control changes in the frame composition required on internodal trunks each time a Class I call is added or deleted.
- The circuit switch function shall interact with the COMSEC function to establish Class I secure calls (end to end encryption).

1.3 PACKET SWITCH FUNCTIONS

The packet service provides for data transmission of packets (or parts of messages). The packet is an independent entity containing all necessary control and routing information to permit forwarding from source to destination hosts. The packet switch will be accountable for individual packet acceptance and delivery but not for total message storage or accountability. Data transfer is near real time in that packets are not delivered thru the network unless they can be accepted by the receiving hosts. The source hosts provide for flow control in that the network commands sources to holdup transmission to the network when the network is too congested to handle the traffic. The packet provides a common format for all asynchronous data communications. The packet switch will also contain Terminal Access Controllers to convert terminal unique interfaces to the standard packet format. The packet medium provides for efficient transmission of asynchronous data since transaction data can utilize more than one route and because bandwidth is not reserved during transmission gaps.

1.3.1 TERMINAL ACCESS CONTROLLER (TAC)

Data terminals will interface directly with the node. Data terminals are defined as character-oriented devices capable of conducting a communication with only one destination at a time. Terminals input to the Packet Switching (PS) network I/A, Q/R bulk or narrative type data traffic. Terminal devices may be computer peripheral controllers and intelligent or unintelligent input/output devices. An example of a computer terminal is a Remote Job Entry Station. Examples of unintelligent terminals are: keyboard/printer (TTY-33, 35, 37), and nonprogrammable buffered CRT's. Examples of intelligent terminals are: ANSI standard buffered CRT, programmable (ADCCP) buffered CRT, magnetic tape/card (ADCCP), and magnetic tape/card (Mode I)

terminals. Terminals will require different access circuit control procedures depending on the individual terminal characteristics. The node will interface with terminals so as to minimize the hardware and software impacts on these users. Programmable Terminal Access Controllers will be employed to serve as interface handlers between specific terminal types and the node main processing system. The terminal access controllers provide the unique access circuit control procedures required of specific terminals and provide the standard host interface looking towards the main processing system of the node.

1.4 NODE CENTRAL CONTROL

The nodal central control function will provide for common nodal functions:

- 1) Node Configuration and Load Control
- 2) Traffic and Routing Control
- 3) Performance Assessment and Status Monitoring
- 4) Fault Diagnoses
- 5) Patching and Testing
- 6) Communication with a Network Control Center

1.4.1 NODE CONFIGURATION CONTROL

Node configuration and load control includes system initialization, automatic replacement of failed equipments, distribution of processing load among the various node processors, automated recovery following equipment repair, and automatic software patching and table updating.

1.4.2 ROUTE CONTROL

Traffic and routing control includes dynamic control of traffic congestion and routing based on the following status data collected:

- a) Line and Trunk Outages in the Network
- b) Estimates of delays anticipated from status data collected
- c) Estimates of Node and Trunk Congestion
- d) Network Topography

1.4.3 PERFORMANCE ASSESSMENT AND STATUS MONITORING

This function includes monitoring of node equipment status, monitoring node performance, development of node traffic statistics, and node fault diagnosis. This function monitors both current status and performance, and status and performance over an extended time period.

1.4.4 FAULT DIAGNOSIS

This function provides the necessary maintenance aid to permit rapid location of equipment failures.

1.4.5 PATCH AND TEST

The patch and test performs the quality assurance monitoring, patching, testing, coordinating, restoring, and reporting functions necessary for effective technical supervision and control over internodal trunk circuits and node access circuits traversing or terminating in a facility.

1.5 DYNAMIC CHANNEL ALLOCATION

A key feature of this node shall be reduction of transmission costs by efficient multiplexing of voice and data on internodal trunks. Data in queue shall, where possible, be transmitted over unused voice slots. Furthermore, since Class I calls can have different data rates it is important to dynamically change call slot locations so that trunk bandwidth is not wasted. For example, if a voice call operating at a 32 KBS rate terminates and a 16 KBS call is initiated in its place, it is important to efficiently utilize the remaining 16 KBS channel. The node design shall permit available bandwidth to be used both for data

delivery and for permitting new Class I calls to be established remembering that a wide mix of Class I rates must be accommodated by the node. Section 2.1 suggests a trunk frame structure which maximizes trunk efficiency.

1.6 NODE INTERFACES

The node will interface with voice and data subscribers and external voice and data networks. Voice subscribers will employ secure and non-secure phones directly connected to the node via access lines which may be several miles to a couple of hundred miles long. Data subscribers which interface the node will include host computers employing Mode VI line protocol and character oriented terminals employing Mode 1, Mode IIA or Mode VI line protocols. Voice Networks which may interface the node include commercial 2 and 4 wire PBX's, TELCO exchanges, AUTOVON, AUTOSECVCOM and TRITAC networks. External Data Networks which may interface the node include other data networks such as AUTODIN II, SATIN IV, WWMCCS, ARPA, etc.

1.7 NODE SIZES

The node design will be highly modular so that different nodes can accommodate different mixes and quantities of voice and data traffic. The approximate range of voice traffic to be accommodated by the various nodes will be 150 erlangs on the low end and 1200 erlangs on the high end. The approximate range of data traffic to be accommodated shall range from approximately .5 MBS on the low end to 5 MBS on the high end.

1.8 TRAFFIC CHARACTERISTICS

A wide variety of traffic types will be handled by the node. Two basic classes of traffic to be accommodated are Class I and Class II traffic. Class I traffic includes real time traffic that cannot be interrupted once started. Typical examples are voice, video and certain forms of facsimile traffic. Class II

traffic includes the general class of store and forward traffic such as high speed interactive traffic, narrative record and bulk data.

1.8.1 TRAFFIC RATES

Class II data traffic to be accommodated can range in data rate from 75 baud to 56 KBS. Class I traffic can range in rate from 2400 KBS to 200 KBS.

1.9 PACKETS VS. SEGMENTS

Basic to the understanding of the study is the distinction between a "segment" and a "packet". In our proposed communication system, messages from a common area are entered via input devices and collected at a common point called the "host". The host breaks down messages into portions, assigns the proper leader heads, and transmits the portions to the first of a series of network nodes. These messages portions transmitted between the host and the first encountered node are called "segments". Upon receiving these segments, the node assigns additional node control information to them. The combination of a leader and node control information is called a leader. A packet, then, is a segment with a leader and is transmitted between nodes. It should be noted that while segments may be of various formats, all packets are of the same format through node processing. In addition, our system converts one segment into one packet, while other systems may convert a segment into many packets.

2.0 DYNAMIC CHANNEL ALLOCATION

This section examines the feasibility and methodology of employing the dynamic channel allocation concept. This concept applies to internodal trunks which interconnect unified nodes. All classes of traffic handled by the node are intended to be efficiently integrated and multiplexed onto the internodal trunks. Figure 2.1 illustrates the illustrative concept provided in the ADPT SOW. RCA studies of the concept have further definitized the frame overhead and have resulted in the recommendation that class III traffic be handled as class II data, thus eliminating this traffic group and its associated marker. Further recommendations suggest that the class I/class II marker be a virtual rather than actual demarkation requiring no transmission overhead. A class II programmable limit is proposed in order to limit the maximum average data rate of class II traffic on the trunk.

The following parts of section 2 provides the rationale and trades which define the recommended frame structure, studies the impact of dynamic channel allocation on system parameters, and recommends an implementation concept.

2.1 FRAME STRUCTURE

The frame structure was examined to determine recommended frame rate, frame size, and frame structure.

The basic tradeoff criteria involved in these studies are transmission efficiency, impact on processing and storage requirements, accommodation of standard class I traffic rates, and traffic delay considerations.

The key conclusions are to use T-1 trunks employing a frame size of 15,440 bits. Since the T-1 rates is 1.544 MBS, the resultant frame period is 10 milliseconds (100 frames per second).

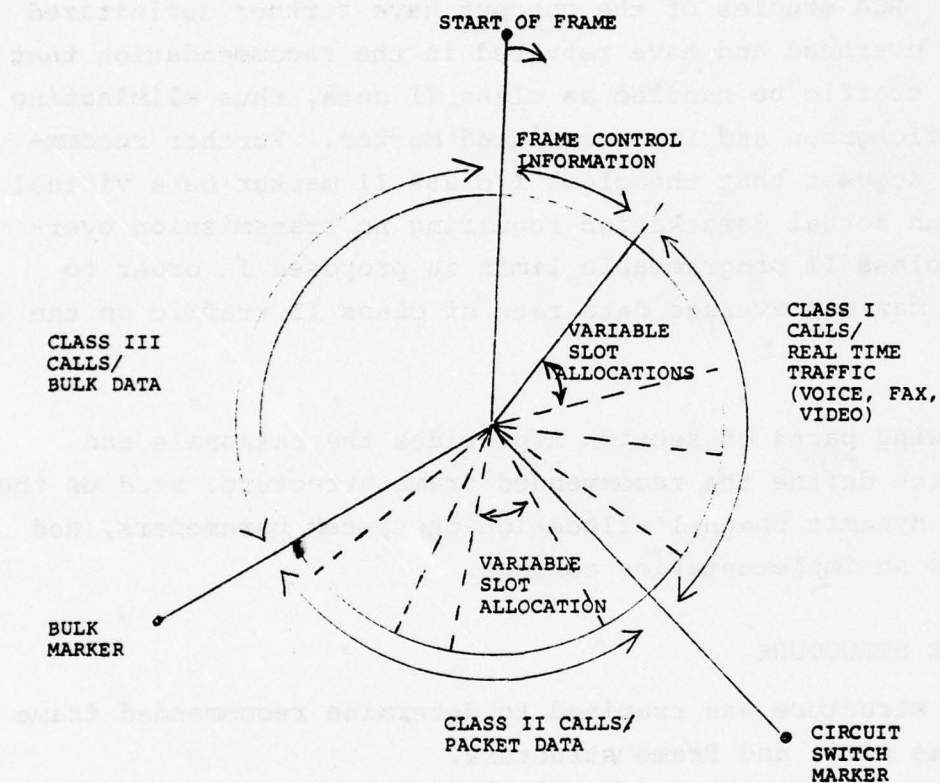


FIGURE 2.1

MULTIPLEX STRUCTURE OF
INTERNODAL TRUNKS

It is further proposed to handle only class I and class II formats. The class I/class II boundary is determined by programmed limits and will not require a special boundary marker within the frame. The class II data shall fill the frame without gaps (the frame can terminate transmitting a partial packet completing transmission of the packet in succeeding frames).

2.1.1 FRAME STRUCTURE BASELINE

Figure 2.1 represents the multiplexed frame structure presented in the statement of work for the ADPT test bed (PR-C-5-2033). This suggested frame structure will be used as a basis for frame study and evaluation. The study will also develop a recommended structure in more detail. Let us first describe Figure 2.1.

In this approach, a frame is some fixed number of bits. The frame is repeatedly sent such that $Br = Nf \times B/Fr$

where Br = Trunk Bit Rate in Bits/
Second

Nf = Number frames per
Second

B/Fr = Bits per frame

The beginning of the frame contains frame control information. Following the frame control in the baseline frame are the class I calls. It is noted that a mix of call rates is accommodated by employing a different slot size for each call in accordance with call bit rate. This not only permits different voice rates to be mixed on the trunk, but also facilitates handling of fax interactive graphics and coded video data, which are also considered class I data (real time synchronous traffic). In the suggested configuration, Class I traffic is compact on the trunk (sent contiguously). A marker shown on the diagram indicates the separation between class I and class II traffic. Since both ends of the link

must be cognizant of the frame structure, and since frame revision is required each time a class I call is added or deleted, there is no need for a real marker to separate class I and class II traffic. Furthermore, a real marker would introduce the complication of marker recognition. The marker is implicit in the frame list agreed upon at both ends of a link, and is therefore changed automatically as the frame list is changed.

Class II calls represent packetized data traffic. This traffic is effectively asynchronous since the packets which are variable length bounded by flags can be queued (delayed).

Figure 2.1 indicates another marker and a third class of traffic (Class III). Section 2.2 of this report examines methods of handling Class III data. The recommendation developed in Section 2.2 is to handle bulk data as a lower priority Class II packet or as Class I call. This recommendation eliminates the Class II/III boundary and results in the recommended revised frame shown in Figure 2.2.

A key recommendation and one which has in fact been implemented in the ADPT test bed is the ability to terminate the frame without completing transfer of the final packet in transit in that frame. The remainder of the interrupted packet is sent in the next frame, following the Class I traffic. If it were not possible to cut up a packet in this manner, inefficiencies would result because the last bit positions in the frame would not always be usable. The implementation approach recommended is a simple one. All Class II traffic received on the trunk is steered to a port of a data memory interface controller which accepts the data on a byte by byte basis using a ready/resume protocol. At the end of the frame, class II data terminate, so no new traffic is routed to this controller.

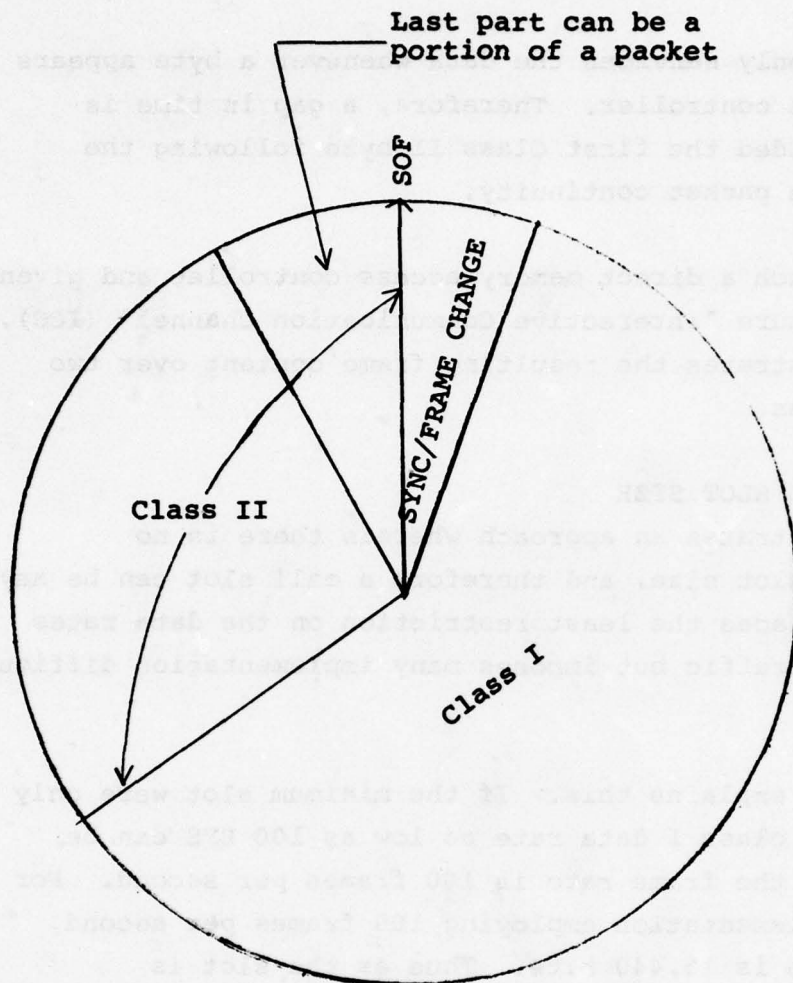


Figure 2.2 Frame w/o Class III Traffic

The controller only services the data whenever a byte appears at a port of the controller. Therefore, a gap in time is acceptable provided the first Class II byte following the gap picks up the packet continuity.

RCA has built such a direct memory access controller and given it the nomenclature "Interactive Communication Channel" (ICC). Figure 2.3 illustrates the resulting frame content over two successive frames.

2.1.2 FRAME AND SLOT SIZE

Figure 2.1 illustrates an approach wherein there is no restriction on slot size, and therefore a call slot can be any length. This places the least restriction on the data rates of synchronous traffic but imposes many implementation difficulties.

An example best explains this. If the minimum slot were only one bit wide, a class I data rate as low as 100 BPS can be accommodated if the frame rate is 100 frames per second. For a T-1 trunk implementation employing 100 frames per second, the frame length is 15,440 bits. Thus as the slot is widened, it is possible to develop a synchronous link operating from as low as 100 BPS or any multiple thereof up to 1.544 MBS. Unfortunately, manipulation of single bits involves much more processing than if a group of bits were to be transferred simultaneously. Since the total real time throughput requirement is high, super high speed processing circuits would have to be employed if processing at the bit level is used. Therefore, it is desirable to determine a word size (bit grouping) greater than one bit, which can accommodate most or all of the standard digital data rates.

In general, the standard digital rates fall into two categories, those that are equal to 75×2^N ($N = \text{Integer}$), and those equal to $8000 \times N$, $N = \text{Integer}$. The following seven standard rates are noted from the SOW:

- 1) 64 KBS
- 2) 56 KBS
- 3) 48 KBS
- 4) 32 KBS
- 5) 16 KBS
- 6) 8 KBS
- 7) 2.4 KBS

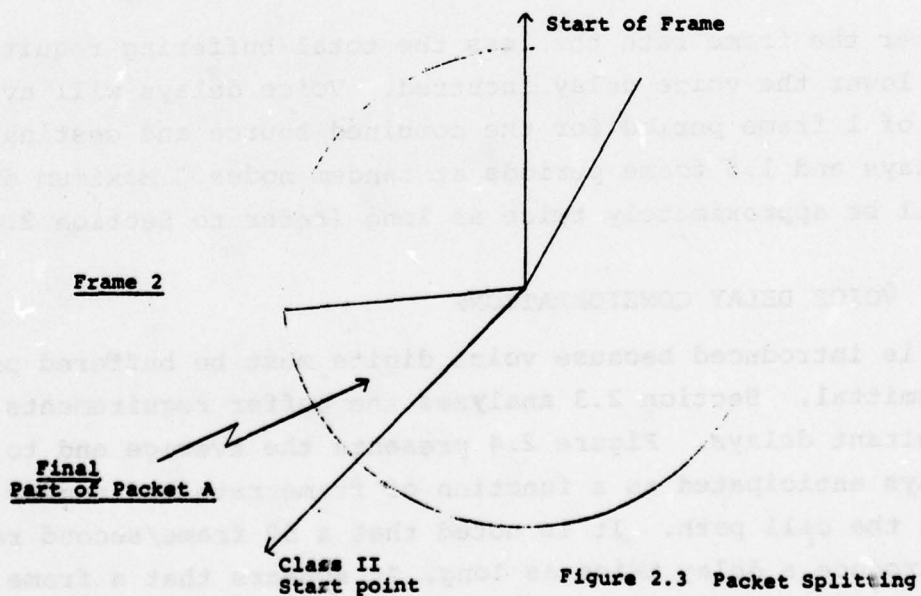
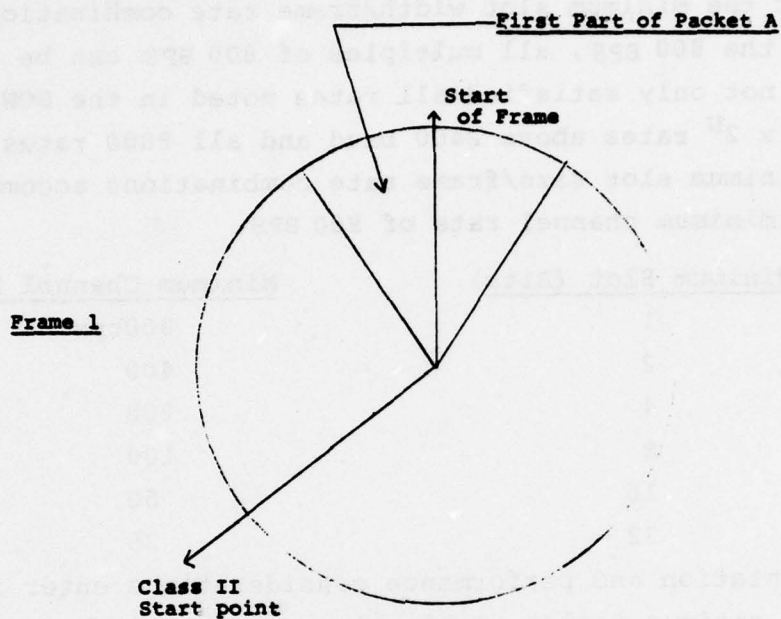


Figure 2.3 Packet Splitting

In addition, 4.8 and 9.6KBS rates have been suggested as future standards for predictive voice encoders.

The common submultiple of all the listed rates is 800 BPS. Furthermore, if the minimum slot width/frame rate combination can accommodate the 800 BPS, all multiples of 800 BPS can be satisfied. This not only satisfied all rates noted in the SOW, but also all 75×2^N rates above 2400 baud and all 8000 rates. The following minimum slot size/frame rate combinations accommodates the desired minimum channel rate of 800 BPS.

<u>Minimum Slot (Bits)</u>	<u>Minimum Channel Rate</u>
1	800bps
2	400
4	200
8	100
16	50
32	25

Both implementation and performance considerations enter into choosing the optimum design combination.

The higher the frame rate the less the total buffering required, and the lower the voice delay incurred. Voice delays will average a total of 1 frame period for the combined source and destination node delays and 1.5 frame periods at tandem nodes. Maximum delays will be approximately twice as long (refer to Section 2.3).

2.1.2.1 VOICE DELAY CONSIDERATIONS

A delay is introduced because voice digits must be buffered prior to transmittal. Section 2.3 analyzes the buffer requirements and resultant delays. Figure 2.4 presents the average end to end delays anticipated as a function of frame rate and number of nodes in the call path. It is noted that a 50 frame/second rate will introduce a delay twice as long. It appears that a frame rate at least as great as 100 frames/second is required. Based

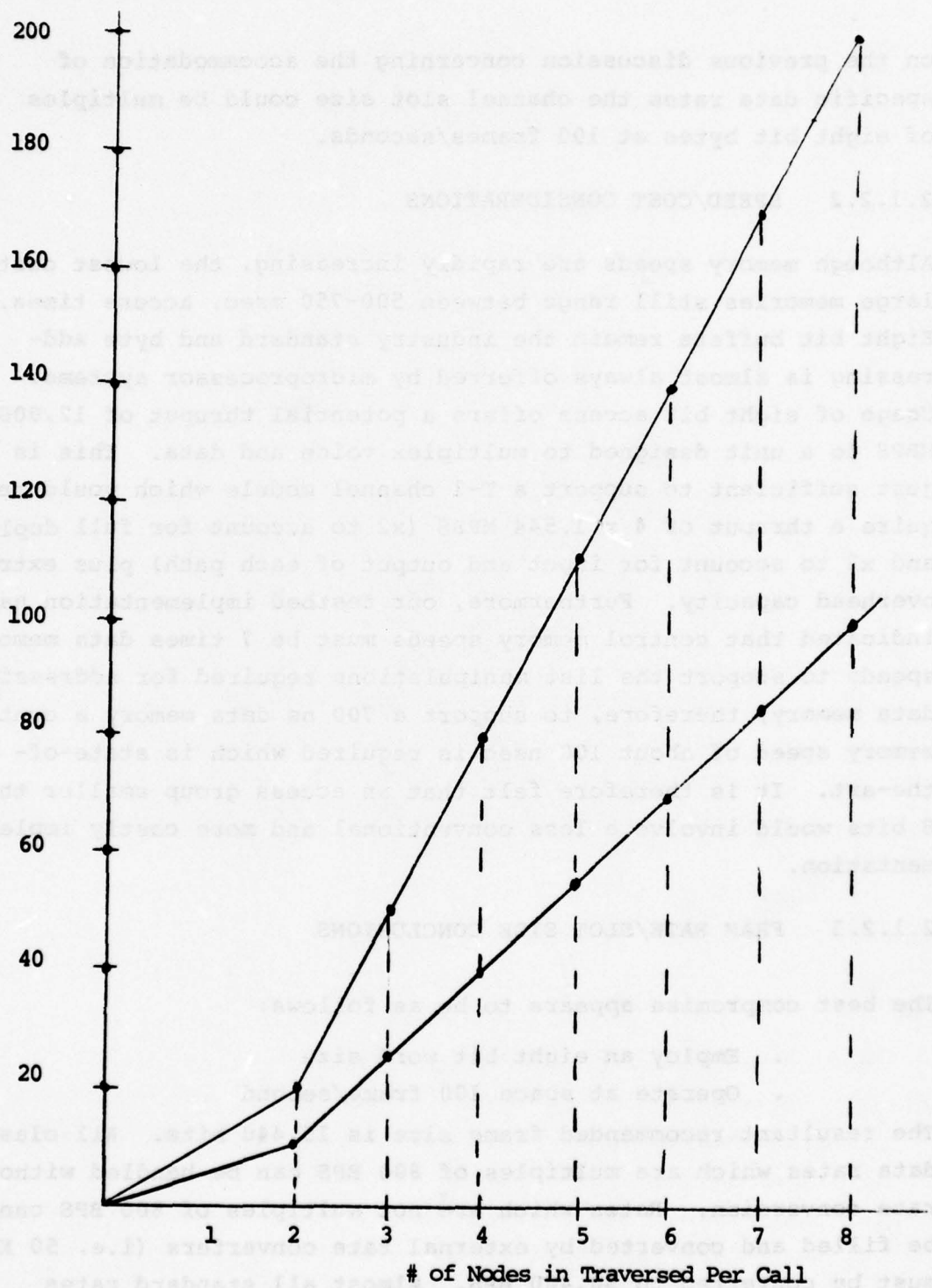


FIGURE 2.4 END TO END CLASS I TRAFFIC DELAYS

on the previous discussion concerning the accommodation of specific data rates the channel slot size could be multiples of eight bit bytes at 100 frames/seconds.

2.1.2.2 SPEED/COST CONSIDERATIONS

Although memory speeds are rapidly increasing, the lowest cost large memories still range between 500-750 msec. access times. Eight bit buffers remain the industry standard and byte addressing is almost always offered by microprocessor systems. Usage of eight bit access offers a potential thruput of 12,000 MBPS for a unit designed to multiplex voice and data. This is just sufficient to support a T-1 channel module which would require a thruput of 4×1.544 MPBS ($\times 2$ to account for full duplex and $\times 2$ to account for input and output of each path) plus extra overhead capacity. Furthermore, our testbed implementation has indicated that control memory speeds must be 7 times data memory speeds to support the list manipulations required for addressing data memory, therefore, to support a 700 ns data memory a control memory speed of about 100 nsec is required which is state-of-the-art. It is therefore felt that an access group smaller than 8 bits would involve a less conventional and more costly implementation.

2.1.2.3 FRAME RATE/SLOT SIZE CONCLUSIONS

The best compromise appears to be as follows:

- . Employ an eight bit word size
- . Operate at space 100 frame/second

The resultant recommended frame size is 15,440 bits. All class I data rates which are multiples of 800 BPS can be handled without rate conversion. Rates which are not multiples of 800 BPS can be filled and converted by external rate converters (i.e. 50 KBS must be converted to 50,400 BPS. Almost all standard rates however can be handled without rate conversion.

2.1.3 SPECIAL FRAME BOUNDARIES

It is deemed necessary and desirable to incorporate two boundaries within the frame structure which limit the areas of occupancy within the frame of Class I and Class II data. A brief discussion of these boundaries follow.

2.1.3.1 CLASS I BOUNDARY

The Class I boundary limits the maximum bandwidth occupancy of Class I data. This serves two purposes:

- 1) The boundary is needed to insure that some of the available trunk bandwidth is dedicated to handling Class II (data packet) traffic. Our analysis and experiments indicate that dedicated data channels are needed to meet delivery delay requirements for interactive data.
- 2) The boundary also modifies the voice grade of service. The fact that the boundary is programmable permits "on line" as well as adaptive modification of the voice grade of service.

The Class I boundary will vary from node to node and will initially be based on estimated node traffic. The Class I boundary will be set up at initialization and can be set to any point in the frame. The Class I boundary can be adjusted to "tune" the system to attain the desired compromise between packet delays and voice grade of service. Although it is beyond the scope of this study to develop a dynamic algorithm to adjust this boundary as a function of the packet data queue lengths, such an algorithm will prove very useful in optimizing total node performance. It is noted that this boundary is determined by the algorithm for accepting or blocking new voice calls and does not require a coded marker within the frame.

2.1.3.2 CLASS II BOUNDARY

The Class II boundary limits the maximum bandwidth occupancy of Class II data. The reason for incorporating this boundary is

primarily due to implementation and cost considerations. The ratio of Class II traffic to Class I traffic during the busy hour can range from about 5-40% depending on node location and the average voice digital rate. In the near term (1980-1985) voice digital rates are likely to remain at or above 16 KBS, but are likely to drop as the developing predictive voice encoder technology advances. Thus, the Class II traffic is likely to be at the low end of the 5-40% range in the near term and the required voice digital thrupt will be at its peak. If the amount of Class II data a node must handle averages only 5% of the total digital traffic during busy hours, then even during peak seconds no more than 12-15% of the available trunk bandwidth would be needed to accommodate this load. DCA statistics do indicate a relative light data load compared to voice load in the near term.

If no limit is placed on Class II data bandwidth, there is always the remote possibility that at times during off hours the total trunk bandwidth would be available for data traffic. This means that the data subsystem must supply data (or idle characters) to each trunk at the full T-1 rate even though this rate is higher than the required Class II data rate (based on actual offered traffic). The cost of the Class II data subnetwork is adversely affected if the worst case thrupt rate is dictated by the total node bandwidth available, rather than the bandwidth required for the Class II traffic. Hence, a programmable limit is recommended which limits the Class II data in the frame. Since the total frame length is fixed it becomes necessary, at times, to fill the class II portion of the frame with "flag" or "idle" characters. If the sum of the actual voice bandwidth occupied and the maximum Class II data permitted is less than the total trunk bandwidth, additional filler bytes are added.

Figure 2.5 illustrates the frame depicting the Class II limit. As is the case for the Class I limit, the Class II limit is a programmable boundary requiring no coded marker with the frame and locatable at any part of the frame.

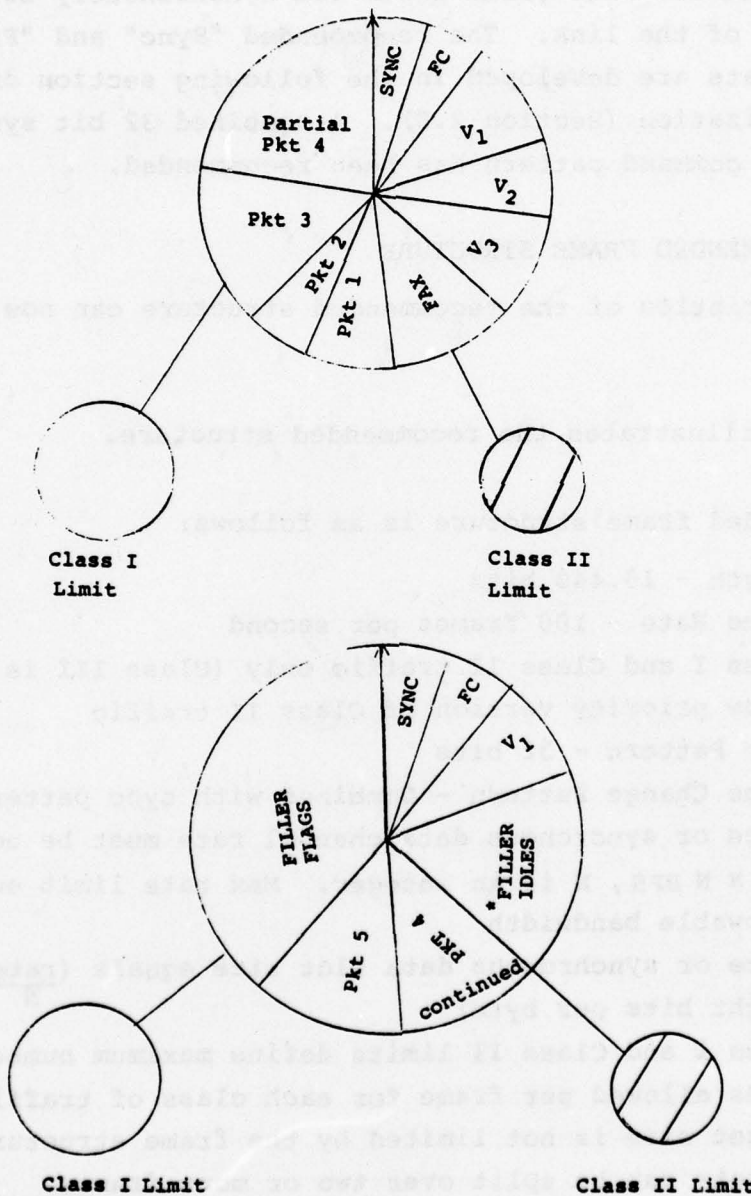


Figure 2.5 FRAMES WITH CLASS II LIMIT

2.1.4 INITIAL FRAME PATTERN

The start of each frame will contain a synchronization pattern and frame change commands. Although CCIS messages could be sent in each frame, the recommended approach is to send CCIS data as packets (see section 3.1). The "Frame Change" commands indicate whether or not the current frame differs from the previous frame. This is a necessary part of the frame composition to assure that frame lists are synchronously switched at both ends of the link. The recommended "Sync" and "Frame Change" formats are developed in the following section on timing and synchronization (section 2.3). A combined 32 bit sync and frame change command pattern has been recommended.

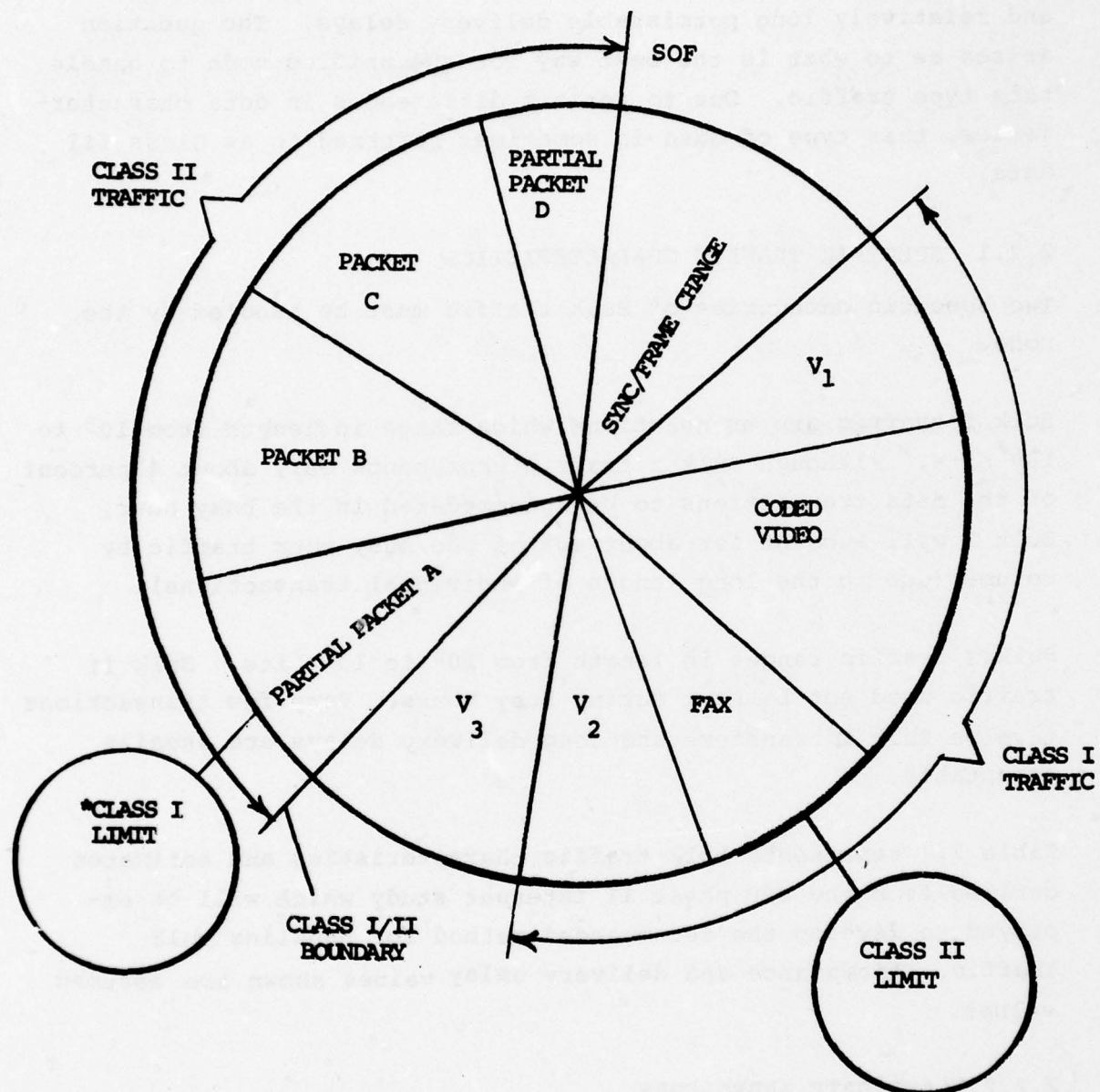
2.1.5 RECOMMENDED FRAME STRUCTURE

The characteristics of the recommended structure can now be summarized.

Figure 2.5A illustrates the recommended structure.

The recommended frame structure is as follows:

- 1) Length - 15,440 bits
- 2) Frame Rate - 100 frames per second
- 3) Class I and Class II traffic only (Class III is handled as a low priority version of Class II traffic)
- 4) Sync Pattern - 32 bits
- 5) Frame Change Pattern - Combined with sync pattern
- 6) Voice or synchronous data channel rate must be equal to $800 \times N$ BPS, N is an integer. Max rate limit equals allowable bandwidth
- 7) Voice or synchronous data slot size equals $\frac{(\text{rate})}{N}$ bytes (eight bits per byte)
- 8) Class I and Class II limits define maximum number of bytes allowed per frame for each class of traffic
- 9) Packet size is not limited by the frame structure since packets can be split over two or more frames.



* NEW CLASS I CALL CAN ONLY BE ESTABLISHED IF SLOT SIZE DOES NOT EXTEND CLASS I TRAFFIC BEYOND PROGRAMMED LIMIT (MARKER).

FIGURE 2.5A RECOMMENDED FRAME STRUCTURE

2.2 BULK DATA HANDLING

A substantial volume of the total data traffic consists of bulk traffic. Bulk traffic is characterized by long message lengths and relatively long permissible delivery delays. The question arises as to what is the best way for the unified node to handle this type traffic. Due to certain differences in data characteristics, this type of data is sometimes referred to as Class III data.

2.2.1 SPECIFIC TRAFFIC CHARACTERISTICS

Two specific categories of Bulk traffic must be handled by the node.

Bulk I traffic are transactions which range in length from 10^5 to 10^6 bits. Although Bulk I traffic represents only about 4 percent of the data transactions to be accommodated in the busy hour, Bulk I will account for about 40% of the busy hour traffic by volume (due to the long length of individual transactions).

Bulk II traffic ranges in length from 10^6 to 10^8 bits. Bulk II traffic need not be sent during busy hours. Very few transactions involve Bulk II transfers and long delivery delays are usually acceptable.

Table 2.1 represents bulk traffic characteristics and estimates derived from the DOD phase II internet study which will be employed to develop the recommended method for handling bulk traffic. Acceptance and delivery delay values shown are assumed values.

2.2.2 CANDIDATE APPROACHES

2.2.2.1 APPROACH 1 - SPECIAL CLASS III SERVICE

The ADPT SOW suggests filling the multiplexed frame transmitted on internodal trunks with bulk data whenever Class I and Class II data does not completely fill the frame. Figure 2.1 illustrates the suggested approach. The approach has the advantage of good transmission efficiency since the trunk is 100 percent utilized during periods where bulk data is available for transmission. One shortcoming of the approach is that once a path is established it is possible that one of the nodes in the path becomes saturated with Class I and Class II data, thus, the end to end throughput rate of bulk becomes limited by the minimum bandwidth available in the path. Furthermore, due to intermediate node throttling, the bulk data stream may dry up during the middle of the transaction. In order to preserve bit integrity it therefore becomes necessary to delineate by special markers in the frame both the start and end positions of the bulk data within the frame. If only one bulk transfer is handled per trunk group, a typical node may handle only a few bulk transfers simultaneously. The net effect as far as Bulk I traffic is concerned is that bulk transaction delivery delay will be composed primarily of acceptance delay. That is to say, the network will not be able to accept the bulk transaction for perhaps several minutes until a trunk path to the destination is found. Actual transmission period following acceptance will be relatively short. This is contrary to good design practice. The acceptance time should be less than one minute.

Another serious drawback is the added hardware complexity required to generate and decode Class III markers. Since these markers would change relative position in the frame almost every frame, the timing involved is critical.

The markers would also have to be fairly long (i.e., 32 bits or larger) to assure detection and would, therefore, represent a significant overhead penalty. A typical situation might be that 600-700 bits of the frame is available (on the average) for Class III data. If a 32 bit start marker and 32 bit end marker is employed per frame, the overhead penalty due to markers alone would be 10 percent.

Table 2.1 - Bulk Data Traffic Characteristics

	AVERAGE TRANSACTION LENGTH	ALLOWABLE ACCEPTANCE DELAY	DELIVERY TIME	AVERAGE NUMBER OF TRANSACTIONS INCIDENT PER AVERAGE PER BUSY HOUR
Bulk I	300 KB	20 sec.	5-20 min.	3200
Bulk II	18 MB	30 sec-min	4-12 hrs.	* 430

* per day

2.2.2.2 APPROACH 2 - HANDLE BULK AS CLASS II SERVICE

If the source computer partitions the bulk data into segments, the bulk data can be transmitted as Class II data. Furthermore, Class II packets, when outputted to a trunk will be sent in accordance with assigned priorities. Assuming bulk data is the lesser priority Class II data, the resultant frame is per Figure 2.6. It is noted that the net effect is that the Class III data occupies the same portion of the frame as originally postulated in approach 1. This approach offers several advantages over approach 1. There are:

- 1) Since packers are asynchronous, no special provisions are needed to maintain bit integrity;
- 2) No special Class III markers are required;
- 3) Many bulk calls can be ongoing simultaneously since packets of different transactions can be queued for the same trunk. Acceptance delay is short being dependent only on buffer availability and not on transmission path availability;
- 4) The bulk traffic is not constrained to a single path between source and destination, but has the same advantage as Class II data in that each packet can reach the destination node via any available path;
- 5) Error control is provided by the network, thus reducing the processing burden on the source and destination hosts;
- 6) Since relatively long delays are permissible per packer, effective use can be made of the bandwidth normally reserved for voice circuits to meet the specified grade of service. (Benefit of Dynamic Channel Allocation).

There are two drawbacks. One disadvantage of this approach is that packet data has an overhead penalty that is relatively high, compared to conventional circuit switching, for those transactions involving long holding times. Bulk data transfers, especially bulk 2, would be transferred more efficiently (less overhead penalty) employing circuit switch techniques.

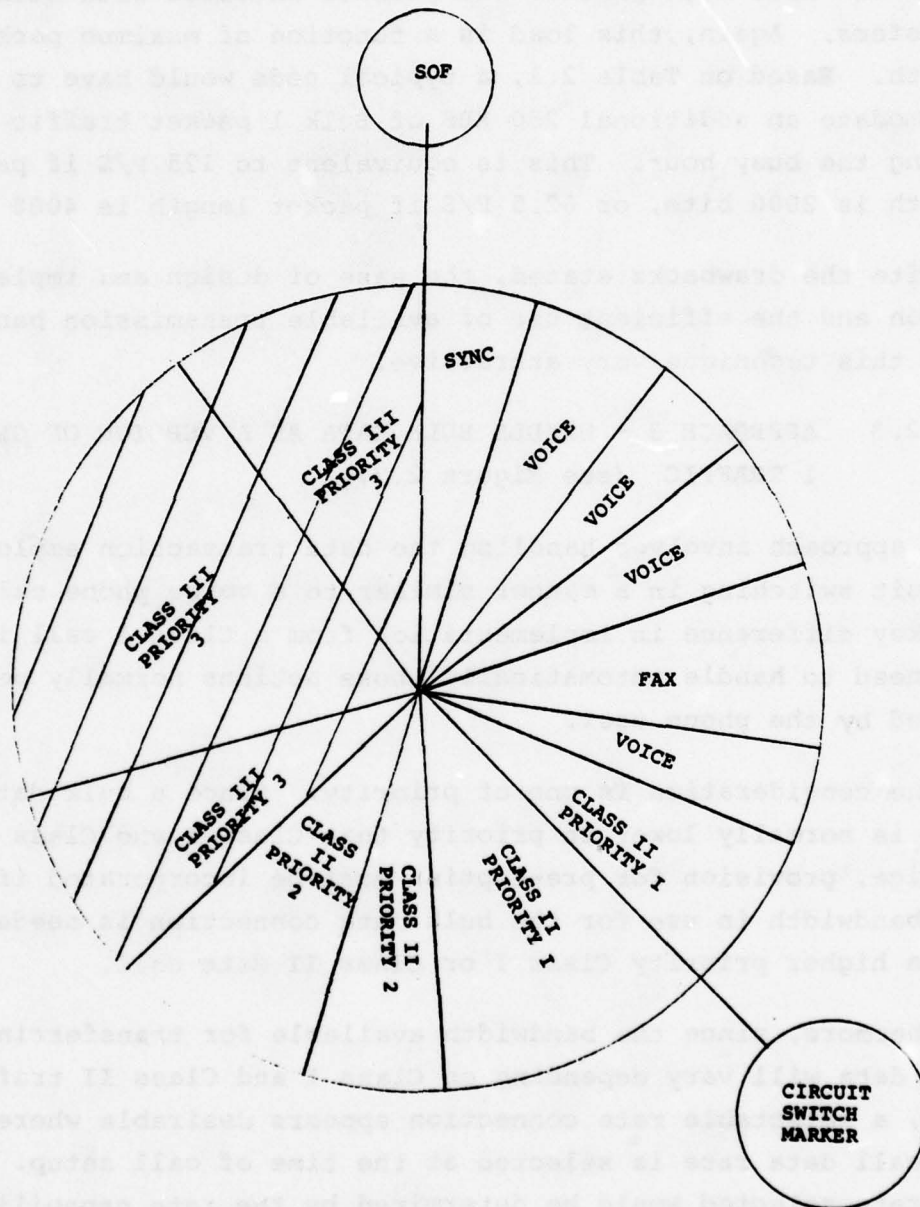


FIGURE 2.6 FRAME CONTAINING CLASS III DATA PACKETS

Typical overhead penalty for handling a data packet 2000 bit long, including acknowledgements, is approximately 17 percent. It is noted that use of longer packets (i.e., 4000 bit lengths) would halve the overhead penalty.

Another disadvantage is the increased packet processing burden, since the node must process the packets involved with Bulk transfers. Again, this load is a function of maximum packet length. Based on Table 2.1, a typical node would have to accomodate an additional 250 KBS of Bulk 1 packet traffic during the busy hour. This is equivalent to 125 P/S if packet length is 2000 bits, or 62.5 P/S if packet length is 4000 bits.

Despite the drawbacks stated, the ease of design and implementation and the efficient use of available transmission bandwidth make this technique very attractive.

2.2.2.3 APPROACH 3 - HANDLE BULK DATA AS A VERSION OF CLASS I TRAFFIC (see Figure 2.7)

This approach involves handling the data transaction employing circuit switching in a manner similar to a voice phone call. One key difference in implementation from a Class I call is the need to handle automatically those actions normally performed by the phone user.

Another consideration is one of priority. Since a bulk data call is normally lower in priority than Class I and Class II service, provision for pre-emption must be incorporated if the bandwidth in use for the bulk data connection is needed for a higher priority Class I or Class II data call.

Furthermore, since the bandwidth available for transferring bulk data will vary depending on Class I and Class II traffic load, a selectable rate connection appears desirable wherein the call data rate is selected at the time of call setup. The rate selected would be determined by the rate capabilities

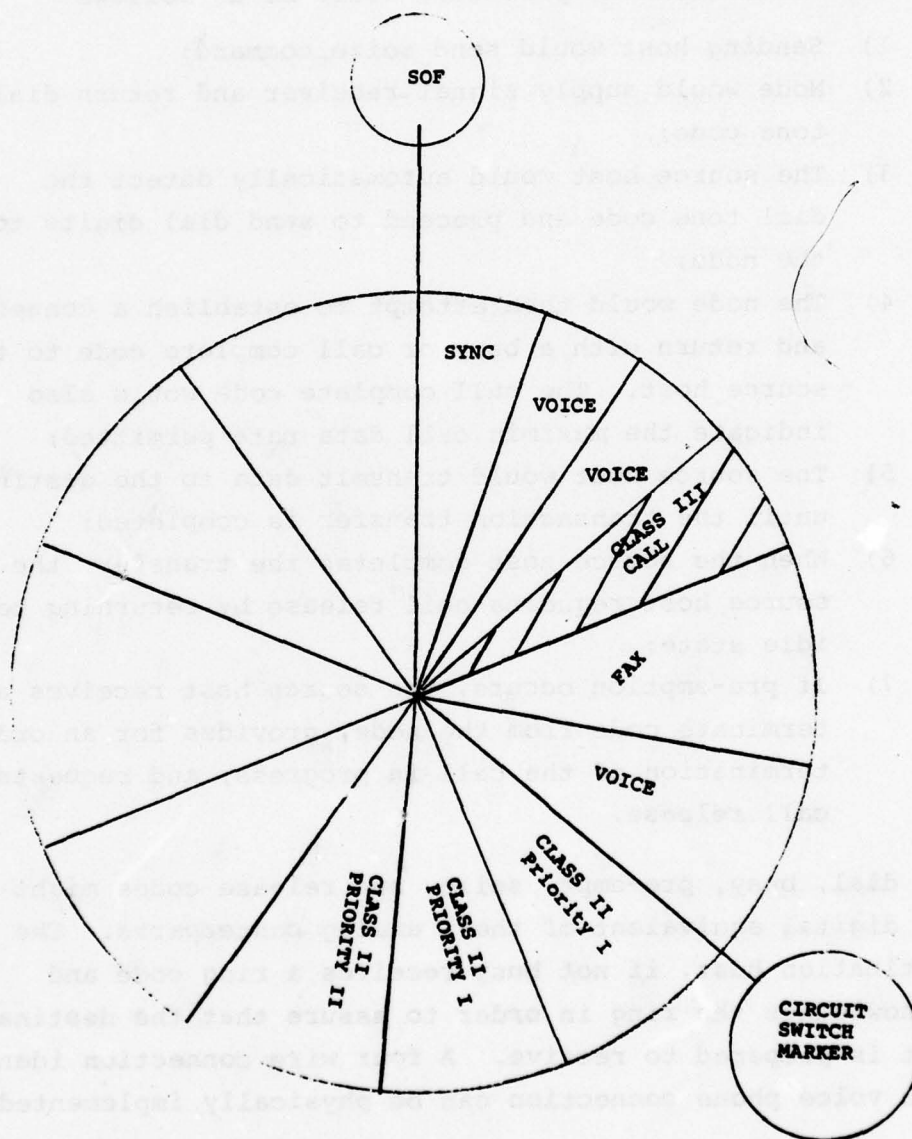


FIGURE 2.7 APPROACH 3 - FRAME CONTAINING CLASS III SYNCHRONOUS CALLS

of the data source and the bandwidth available for establishing the end to end link.

The link established would be a one way rather than full duplex trunk link, however, the loop connection would be full duplex for signalling and supervision interchanges.

The call establishment procedure would be as follows:

- 1) Sending host would send seize command;
- 2) Node would supply signal receiver and return dial tone code;
- 3) The source host would automatically detect the dial tone code and proceed to send dial digits to the node;
- 4) The node would then attempt to establish a connection and return either a busy or call complete code to the source host. The call complete code would also indicate the maximum call data rate permitted;
- 5) The source host would transmit data to the destination until the transaction transfer is completed;
- 6) When the source host completes the transfer, the source host requests call release by returning to the idle state;
- 7) If pre-emption occurs, the source host receives a terminate code from the node, provides for an orderly termination of the call in progress, and requests call release.

The dial, busy, pre-empt, seize, and release codes might be the digital equivalent of their analog counterparts. The destination host, if not busy receives a ring code and acknowledges the ring in order to assure that the destination host is prepared to receive. A four wire connection identical to a voice phone connection can be physically implemented.

The key advantage of this approach are:

- 1) Via use of a "Data Adapter" a terminal may be provided with dual capability (voice and data service).
- 2) The implementation of this function in the network node is relatively simple. The implementation involves classmarking the lines which have data capability and generating and detecting those patterns unique to the data call. The software algorithms for call establishment and disestablishment are also impacted. A key problem is the development of the algorithms for determining how much bandwidth to allocate for the data call and the development of a pre-emption algorithm which prevents Class II data delays from becoming excessive.
- 3) The overhead penalty is less than other approaches. The only overhead involves the call establishment and releases messages. No overhead data is needed during the actual call. Pre-emptions will, of course, increase overhead (since this type of call is the lowest precedence, it would be subject to pre-emption more frequently than routine Class I traffic). No special frame markers are needed since the bulk call is handled on the trunk in the same manner as any other Class I call (e.g. voice).

The key disadvantage is the inability to guarantee efficient utilization of available bandwidth. Then a call is established the call bandwidth cannot use all the remaining available bandwidth in the frame since the call would be very vulnerable to pre-emption, therefore, only a percentage of the remaining bandwidth can be utilized. Furthermore, if more bandwidth should become available during the data transfer, it may not be feasible to re-establish the call to operate at a higher data rate.

Since Class I traffic uses a fixed call patch while Class II messages are packetized and can use multiple transmission paths, approach 3 the call to operate at a higher data rate.

Since Class I traffic uses a fixed call path while Class II messages are packetized and can use multiple transmission paths, approach 3 (and also approach 1) is not as efficient in utilizing available bandwidth as approach 2.

Another disadvantage is that Class III data would be transmitted even through higher priority Class II data may be awaiting transmission. (In this approach, Class II overrides Class III only when the backlog of Class II data for the trunk is sufficiently high to cause pre-emption of the Class III call).

2.2.3 EVALUATION AND RECOMMENDATION

Table 2.2 presents a comparison of the various approaches considering a number of criteria which have been discussed in previous paragraphs.

The rationale for the relative ratings for each criteria in Table 2.2 have been discussed previously; however, further clarification of each of the criteria in the table follows:

Implementation Complexity - Impact on node hardware and software. Does not include error control complexity required at sources for approaches 1 and 3.

Transmission Efficiency - Relative ability of each technique to fill available transmission bandwidth with bulk data awaiting transmission.

Overhead - Requirement for extra housekeeping data over and above the actual source data (i.e. messages, markers, leaders, trailers).

Network Error Control - Ability inherent in the approach for detecting data errors which occur within the nodal transmission network.

Priority Handling - Ability of the approach used to transmit the bulk data in accordance with priorities based on individual transaction precedences. Also considers capability of throttling bulk data when higher priority Class I or Class II data is awaiting transmission.

Crypto Synchronization - Primarily concerned with maintaining end to end bit integrity which requires definitive start and stop marker for each continuous transmission.

TABLE 2.2 BULK DATA HANDLING APPROACH COMPARISONS

	IMPLEMEN- TATION COMPLEXITY	XMISSION EFFICIENCY	OVERHEAD	NETWORK ERROR CONTROL	PRIORITY HANDLING	CRYPTO SYNC	MULTIPLE TRANS. TRUNK
Approach 1	Complex	Good	Fair	No	Medium	Difficult	Difficult
Approach 2	Added Packet Load	Best	*9-18%	Yes	Best	Same as Class II	No Problem
Approach 3	Software	Poorest	Best	No	Poorest	Same as Class I	Complex Algorithm

* Depends on Packet Length

Multiple Transmission/Trunk - Ease with which approach can simultaneously accommodate many bulk transactions. It is preferable to transmit many transactions in parallel at a slower effective transmission rate than to transmit transactions serially at high transmission rates, but longer acceptance delays.

2.2.4 CONCLUSIONS

Actual transmission efficiencies cannot be calculated since they are dependent on the connectivity of the total network. However, it is expected that the better transmission efficiency of approach II (packet transmission) will more than compensate for the overhead penalty. In other words, it is expected that approach II will be the fastest means of transmitting bulk data. Approach II appears to be the best approval overall. Approach II also offers network error control. This feature will remove the burden of error control from the source and destination computers resulting in a large overall cost savings to the military. Approach II also does not impact nodal design (although this approach does increase the packet processing load).

It may also be worthwhile to incorporate approach III as a user option (requires addition of nodal software) so as to accommodate dual voice/data terminals. It is noted that both approach II and approach III can coexist in the node on a non-interfering basis.

A hybrid interface is also suggested for those computer users which are expected to interchange large data bases. These users could be dual users of the node. These users would normally be connected to a regular Class II port and would act as a Class II user. By means of segment exchange, a circuit switched connection can be established through the network and the users connected to Class I interface ports for normal exchange of Class I traffic. Either end user, by returning to the idle (release) condition would cause breakdown of the Class I connection. The users would then be switched back to the Class II ports of their respective

nodes. Only very few subscribers per node would be candidates for this special hybrid category of service.

3.2.4 CONCLUSIONS

Actual transaction efficiencies cannot be calculated since they are dependent on the connectivity of the total network. However, it is expected that the better transmission efficiency of approach II (packet transmission) will more than compensate for the overhead penalty. In other words, it is expected that approach II will be the fastest means of transmitting bulk data. Approach II appears to be the best overall overall. Approach II also offers network error control. This feature will remove the burden of error control from the source and destination computers resulting in a large overall cost savings to the military. Approach II also does not impact node design. Although this approach does increase the packet processing load,

it may also be worthwhile to incorporate approach III as a near option (perhaps addition of node software) so as to accommodate both voice/data terminals. It is noted that both approach II and approach III can coexist in the node on a non-interfering basis.

A hybrid interface is also suggested for those computers which are expected to interchange large data blocks. These users could be dual users of the node. These users would initially be connected to a regular Class II port and would act as a Class II user. By means of segment exchange, a circuit switched connection can be established through the network and the users connected to Class I interface ports for normal exchange of Class I traffic. Either end user, by returning to the idle (released) condition would cause breakdown of the Class I connection. The users would then be switched back to the Class II ports of their respective

2.3 TIMING AND SYNCHRONIZATION

This section of the report deals with the problems of timing and synchronization associated with the dynamic channel allocation study. The scope of the study is limited to those aspects of timing and synchronization associated with Dynamic Channel Allocation.

It is not the intent of this study to examine the problem of keeping the clocks at network nodes synchronized. The techniques of master/slave clock schemes, frequency averaging, use of ultra accurate and stable independent clocks and other approaches have all been explored in depth in studies of other switch networks.

This section will examine the following aspects of Timing and Synchronization:

1. Synchronization of the Basic Multiplex Frame;
2. Maintaining synchronization of the frame during dynamic channel changes.
3. Maintaining bit integrity on the synchronous data channels.

2.3.1 SYNCHRONIZATION OF THE BASIC FRAME

The internodal trunk will operate at the T-1 rate and will be formatted in frames. The recommended frame length is 15,400 bits. The resultant frame repetition rate is 100 frames per second (developed in previous section). Let us now determine the characteristics of the sync pattern required to de-limit the frame boundary and the rules for sync acquisition and recognition of sync loss.

2.3.1.1 SYNC ACQUISITION

Two statistical values of probability are important associated with acquisition of frame sync.

- 1) P_a (Probability of Acquisition) Probability that frame sync will be acquired within some specified time period;

- 2) Pfa (Probability of False Acquisition) Probability that the acquired sync position is not the correct position.

2.3.1.2 SYNC LOSS DETECTION

Two statistical values of probability are important associated with detection of loss of sync.

- 1) Pd (Probability of Detection) Assuming an "out of sync" condition occurs, the probability of detecting the "out of sync" condition within a specified time period;
- 2) Pfd (Probability of False Detection) Assuming sync is correct, the probability of falsely detecting an "out of sync" condition (due to bit errors) thus reinitiating the acquisition cycle.

2.3.1.3 CONTROLLABLE VARIABLES

The following parameters affect the previously discussed probabilities:

- N - Number of bits in the sync pattern
- K - Allowable errors in the sync pattern
- Ta - Number of trials that must succeed in sync pattern detection before sync is locked
- Td - Number of trials that must not succeed in sync pattern detection before "out of sync" is declared detected.

The above parameters will be selected so as to meet the required probabilities with an optimum sync pattern.

2.3.1.4 BIT ERROR RATE ENVIRONMENT

The ability to acquire, and the ease with which sync is lost is dependent on the link BER. Typical high quality T1 and satellite links attain BER's in the range of 10^{-5} - 10^{-6} . However, to provide adequate safety margin a worst case BER of 10^{-4} is assumed for the condition of random errors. Burst noise will be a factor over microwave and satellite links. It is assumed that such bursts result in an error rate of 10^{-2} during the burst duration.

2.3.1.5 EQUATIONS OF PROBABILITY

The probabilities of acquisition and sync loss detection are defined by the formulas in Table 2.3. These formulas are readily derived employing the binomial expansion approach and appear in slightly different format in reference 1.

Reference 1 Group Synchronizing of Binary Digital Systems
R. H. Barker

2.3.1.6 DEFINITION OF REQUIREMENTS AND CONSTRAINTS

In order to develop sync characteristics and rules, it is necessary to define a set of performance objectives.

First let us consider the requirements dictated by the normal error environment ($BER = 1 \text{ in } 10^{-4}$).

The following set of acquisition and dropout criteria appear reasonable:

$$P_a \geq .999$$

$$P_{dd} \geq .999$$

$$P_{fa} \geq 10^{-7}$$

The value of P_{fd} will be specified for the burst environment ($BER = 10^{-2}$)

$$P_{fd} \leq 10^{-12}$$

A tight P_{fd} is specified for the low BER for the following reason. Sync is normally detected in the benign environment. Since it is usually much easier to maintain sync once located, the P_{fd} is specified to maintain sync during noise bursts.

One of the parameters which aids in achieving the low P_{fd} is allowing a large number of sync misses before dropout is declared, and a new sync search is initiated (large T_d). A large T_d also guarantees that short noise bursts will not cause

Table 2.3. Formulas for Probabilities
of Interest

$$1. \quad P_a = \left(p^N + C_1^N p^{N-1} q + \dots + C_K^N p^{N-K} q^K \right) T_a$$

$$2. \quad P_{fa} = \left[\frac{\left[15,440 - 2N \right] \sum_{X=0}^K C_X^N}{2^N} \right] T_a$$

$$3. \quad P_{dd} = \left[1 - \left(\sum_{X=0}^K C_X^N / 2^N \right) \right] T_d$$

$$4. \quad P_{fd} = \left[1 - \left(p^N + C_1^N p^{N-1} q + \dots + C_K^N p^{N-K} q^K \right) \right] T_d$$

$$\sum_{x=0}^K (C_x / 2^N) = \text{Probability of random bit stream being detected at sync}$$

$$1 - (p^N + C_1^N p^{N-1} q + \dots + C_K^N p^{N-K} q^K) = \text{Probability that sync pattern in frame is not detected}$$

Table 2.3 (Cont'd)

P_a	= Probability of Sync Acquisition
P_{fa}	= Probability of False Sync Acquisition
P_{dd}	= Probability of Sync Dropout Detection
P_{fd}	= Probability of False Sync Dropout
P	= Probability of data bit being received correctly
Q	= Probability of data bit being received incorrectly ($P + Q = 1$)
N	= Sync Code Length
K	= Allowable errors in sync code
T_a	= Number of sync trials prior to declaring sync acquired
T_d	= Number of sync trials prior to declaring sync dropout

$$C_X^N = \frac{N!}{(N-X)!X!}$$

dropout. The penalty, however, for ignoring sync misses over many frames is that if a true sync loss does occur, recovery time is lengthened. These considerations lead to the definition of two additional constraints.

1. Sync recovery time must be within 150 milliseconds with .99 confidence in the benign environment ($BER = 10^{-4}$). This requirement stated in another manner says that total number of frames for detection and reacquisition shall not exceed 15 frames 99% of the time. An interrupt of 150 millisecond, occurring very infrequently, would only blip on-going voice conversations and not cause great inconvenience.
2. Acquisition time shall be no greater than 20% of the sync lock-in period. This states that if a noise burst must be 100 milliseconds long to drop sync (sync is not detected in ten consecutive frames, the time for reacquisition shall be no more than 20 ms (two frames). As stated previously P_a shall be equal or greater than .999 in the benign environment ($BER = 10^{-4}$).

The overall objective is to determine a combination of controllable variables (sync length, allowable errors in sync pattern, number of trials before "lock-up" or "dropout") which satisfies the above defined requirements and constraints with the lowest penalty in overhead and implementation complexity.

2.3.1.7 SYNC RECOMMENDATIONS

A computer runoff computed the pertinent probabilities for a wide range of controllable variable inputs. Table 2.4 represents some of the computed results. Based on a direct examination of the computed results the following sync pattern length and rules are recommended:

- Pattern Length (N) - 32 bits
- Allowable Errors (K) - 1
- # Trials for Acquisition (T_a) - 2
- # Trials for Dropout (T_d) - 10

Table 2.4 - Calculated Probabilities

LENGTH N	K	#TRIAL VAL	P _a	P _{fa}	P _d	P _{fd}
8	0	1	.9992	1	.996	.0008
		2	.9984		.992	(4 x 10 ⁻⁶)
		4	.9968		.984	36 x 10 ⁻¹²
	1	1	.9999994			
		2	.9999988			
		4	.9999976			
16	2	1	≈ 1			
		2	≈ 1			
		4	≈ 1			
	0	1	.9984	.23.5%	.999985	.0016
		2	.9968	.055	.9997	2.56 x 10 ⁻⁶
		4	.9936	.003	.9994	.4 x 10 ⁻¹²
	1	1	.9999976		.99974	.0000024
		2	.9999952		.99948	5.76 x 10 ⁻¹²
		4	.9999904		.99896	NEG
	2	1	≈ 1			
		2	≈ 1			
		4	≈ 1			
24	0	1	.9976	.0009	1	.0024
		2	.9952	81 x 10 ⁻⁸	1	5.76 x 10 ⁻⁶
		4	.9904	NEG	1	.33 x 10 ⁻¹²
	1	1	.9999944	.023		.0000066
		2	.9999888	.0005		43 x 10 ⁻¹²
		4	.9999776	25 x 10 ⁻⁸		NEG

Table 2.4 - Calculated Probabilities (Cont'd)

LENGTH N	K	#TRIAL VAL	P _a	P _{fa}	P _d	P _{fd}
24	2	1	.999997			
		2	.999994			
		4	.999988			
32	0	1	.9968	.0000035	≈ 1	.0032
		2	.9936	.12 x 10 ⁻¹¹	≈ 1	10.24 x 10 ⁻⁶
		4	.9872	NEG	≈ 1	10.48 x 10 ⁻¹²
	1	1	.99999	10 ⁻⁴	≈ 1	10 ⁻⁵
		2	.99998	10 ⁻⁸	≈ 1	10 ⁻¹⁰
		4	.99996	10 ⁻¹⁶	≈ 1	10 ⁻²⁰
	2	1	.9999948	.00189	≈ 1	.5 x 10 ⁻⁵
		2	.9999896	.0000005	≈ 1	.25 x 10 ⁻¹⁰
		4	.9999792	12 x 10 ⁻¹²	≈ 1	.06 x 10 ⁻²⁰

In a bit error environment of 1 error per 10^4 bits the above parameters result in the following probability values:

Pa -- .99999
Pfa -- $.14 \times 10^{-7}$
Pd -- .9999999
Pfd -- $.88 \times 10^{-53}$

In a bit error environment of 1 error per 10^2 bits the above parameters result in the following probabilities:

Pa ---- .92
Pfa ---- $.14 \times 10^{-7}$
Pd ---- .9999999
Pfd ---- 1.2×10^{-12}

The 32 bit pattern was chosen because it was the smallest eight bit multiple which satisfied the defined requirements. A sync pattern which is a multiple of eight bits is consistent with the byte access philosophy recommended for all channels of the master frame. (A 24 bit pattern is too small to satisfy the acquisition requirements. If one or more errors are permitted in a 24 bit pattern, the probability of false acquisition increases beyond the defined requirement. If no errors were permitted in a 24 bit pattern the acquisition probability drops below the defined requirement.)

It would be possible to employ a 32 bit pattern to acquire sync and a smaller pattern (16 or 24 bits) to maintain sync. However, use of separate sync lengths for acquisition and maintenance introduces considerable complexity in link coordination commands and requires that the return link of the duplex link be operating at a low BER at the time reacquisition is required, so that the transmit end can be instructed to send the longer sync for reacquisition. The added complexity and risk does not justify the overhead savings which is approximately .1% of the total transmission bandwidth.

2.3.1.8 USE OF TWO SYNC PATTERNS

For reasons to be discussed later in this section two complementary sync patterns are recommended for use. Since the number of acceptable sync codes double, probabilities Pfa and Pdd will be adversely affected. Pa and Pfd are not affected. However, employing the same sync length and rules previously recommended, the resultant probabilities are still acceptable. The probabilities adjusted for two sync codes are:

For BER 1 in 10^{-4}

Pa .999999

Pfa 5.6×10^{-8}

Pdd .99999998

Pfd $.88 \times 10^{-53}$

For BER 1 in 10^{-12}

Pa .92

Pfa 5.6×10^{-8}

Pdd .99999998

Pfd 1.2×10^{-12}

2.3.1.9 SYNC PATTERN

The following considerations are important for sync pattern selection:

1. The sync pattern should avoid those patterns most likely to appear in the data stream. These include:
 - . All patterns with five or more contiguous "ones", since these appear as idle, flag, or abort characters in the data stream.
 - . All patterns with evenly alternating ones and zeroes such as
10101010-----or
11001100-----
since such pattern represent D.C. levels or quiescent levels in voice patterns.
2. The pattern should exhibit good autocorrelation characteristics

3. The pattern should not resemble any of the crypto sync patterns

A great many patterns can satisfy the defined sync pattern requirements. Since the code is 32 bits in length, optimum codes with perfect autocorrelation properties are not possible, however, certain codes which are permutations of the Barker codes exhibit good autocorrelation characteristics. An alternative to using a Barker code expansion is to expand the 31 bit maximal length code (generated by cycling a five bit shift register) to 32 bits, since maximal length codes also exhibit good autocorrelation patterns.

One pattern which appears to satisfy the requirements is the following:

00011010010000010101110110001111

It's autocorrelation pattern is as indicated below:

Bit Overlap	32	31	30	29	28	27	26	25	24	23	22
Correlation	32	1	2	-1	-2	-1	2	-5	0	1	-2

Bit Overlap	21	20	19	18	17	16	15	14	13	12	11	10
Correlation	7	0	1	4	1	-2	-1	-4	-5	0	-3	-2

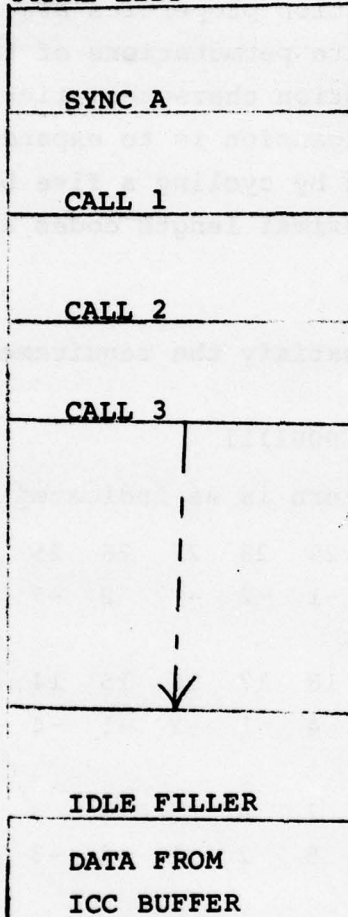
Bit Overlap		9	8	7	6	5	4	3	2	1
Correlation		-5	0	5	2	1	-2	-3	-2	-1

It is recommended that the implementation approach permit the sync pattern to be modified by a software or hardware patch to facilitate test and evaluation of the sync pattern.

2.3.2 TRUNK FRAME SYNCHRONIZATION

The trunk will send and receive a time division multiplexed mixture of Class I and Class II data. The basic frame assigns time slots of variable size for each component of the frame. Figure 2.8 illustrates the basic type of source list used to generate the frame. The list indicates each channel in sequential

FRAME LIST



Contains Byte Size
 Assignment to corresponding
 channel slot

Figure 2.8 TRUNK FRAM LIST

order and indicates the number of bytes required for each channel slot. The final entry on the list calls for sufficient data from the ICC buffer to complete the total frame. Each end of a link must work with identical lists and list changes must be synchronized. The proposed method is to employ two lists per trunk, let's call them list "A" and list "B". These lists are alternated for changes. Normally, one list is being used as the "on line" list and the other list is in a standby mode. When a list change is required, the standby lists are correspondingly updated at each node connecting the trunk. Since the lists being updated are not in current use, updating of the list at each node end need not be synchronous and can be coordinated between node ends via a packet message exchange. The next step is to switch the updated standby lists "on line" without interruption of the bit stream. The switchover must occur synchronously and at the start or endpoint of the list.

It is proposed to use two sync patterns to command frame changes. One of two complementary sync patterns is transmitted every frame. If the next frame is to employ list A, then sync pattern "A" is transmitted. If the next frame is to employ list B, sync pattern "B" is transmitted. Thus, the receive end always knows which list is to be employed during the following frame. The frame is changed only when voice or other Class I calls are added or deleted.

A master/slave arrangement is suggested (to avoid glare) wherein frame changes are always initiated at the master end. The term "frame change command" is used to signify the command to switch lists. It is recommended that sync A be used to command selection of the A frame list and sync B selection of the B list. This eliminates the need for special independent commands. Figure 2.9 illustrates the frame change sequence. The following steps are involved in a single frame change:

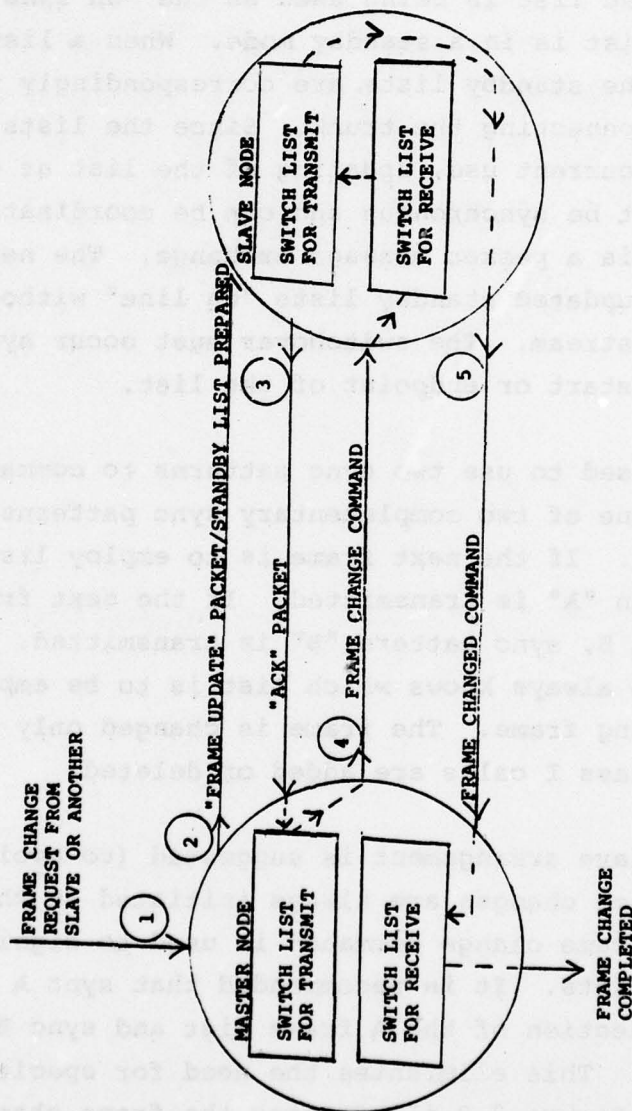


Figure 2.9 FRAME CHANGE SEQUENCE

1. Master end of Link is notified via the slave end, or another link, that a frame change is required on a particular link.
2. Master sends a "frame update" packet to the slave node and prepares its own new standby frame list.
3. The Slave Node acknowledges receipt of the packet by sending a "ACK" packet back to the Master after the new frame list is placed in standby at the slave end.
4. The Master sends a frame change command (alternate sync pattern) and switches its list for transmit.
5. Slave receives the frame change command and switches its list for receive.
6. Slave sends frame change command on return leg of trunk and switches its list for transmit.
7. Master receives frame change command on return leg and switches its list for receive.
8. Each end updates its alternate frame list such that both "A" and "B" lists are identical.

It is noted that by prearrangement the single frame change can add or delete several calls simultaneously provided the frame update packet specifies all list changes required.

In order to insure that frame synchronization is a solid procedure the following precautions are recommended:

1. The sync patterns are complementary. The sync decision (as to which frame is commanded (A or B)) should be made by a best fit decision, thus at least 16 of the 32 bits must be in error to incorrectly interpret sync. This is highly improbable even in a high noise environment.
2. The list taken "off line" after a frame change becomes the new standby. The list should be updated immediately following the frame change to be made equal to the "on line" list, thus even if a subsequent frame change command

(sync pattern) is not properly decode, use of the alternate list will not normally perturbate the trunk bit sequence.

3. The master node shall monitor that the proper sync code is received from the slave node after swichover is commanded to insure that the slave node has switched, prior to initiating new swichover commands.
4. Both the master and slave node should inhibit frame changes during the interval when its standby list is being revised.
5. The master node should not accept any frame change commands from the slave except those originally initiated by the master.
6. Frame change decoding should be inhibited at any time "out of sync" is indicated on the receive trunk.
7. An error packet message should be generated by the slave and sent to the master if frame change commands are alternated without prior notification via the CCIS packet exchange, in which case the master shall reinitiate the last list change command.

Although the above rules will insure that lists track each other, and insure synchronous list switching, there is always a remote possibility that lists are not tracking (i.e. due to internal transients). A background test and recovery mode is, therefore, necessary. In this mode, the slave periodically (i.e. over per minute) sends its current list to the master for cross-checking. The master keeps sufficient history data to permit checking of the list allowing for recent list changes

which may not be incorporated in the received list. If the list contains disagreements, a new list is prepared by the master purging those calls which are not agreed upon at both ends of the link. Call release packets are generated as required to breakdown any calls listed at only one end of the link.

2.3.3 MAINTAINING BIT INTEGRITY

It is very important to maintain bit integrity from end to end when connecting secure phones. If a bit slip occurs, crypto synchronization is lost. Each node contains a buffer for each Class I call in progress. The total size required for this buffer varies for each type call as a function of call data rate and call routing and must be sized to prevent loss or gain of bits.

Figure 2.10 illustrates the flow of a call from source to sink thru four nodes. The key to maintaining bit integrity is by two means:

1. At call setup time, the bit distance is maximized between the point of buffer input and point of buffer output. The object is to prevent overflow wherein read-in data catches up to data read out, or underflow, wherein data read-out catches up to read-in point.
2. The buffer allocated must be large enough to prevent overflow or underflow.

2.3.3.1 BUFFER SIZE

Several factors affect buffer size. These are:

1. Clock differenced between adjacent nodes. Incoming data on a trunk is usually clocked by a clock derived from the received data while outgoing data is clocked by the local clock.
2. Speed Buffering - Since loop and trunk rates are not the same, the buffer is necessary to accommodate the rate differences (e.g. Voice line may operate at 16KBS while voice data is transferred onto and off the inter-nodal trunk at the T1 rate).
3. The affect of dynamic channel allocation on trunks. Since the slot position of a voice call will vary within the frame (due to slot rearrangements as voice calls are added or delated) the point in time data enters or leaves the trunk varies, thus requiring buffering to accommodate the delays introduced.

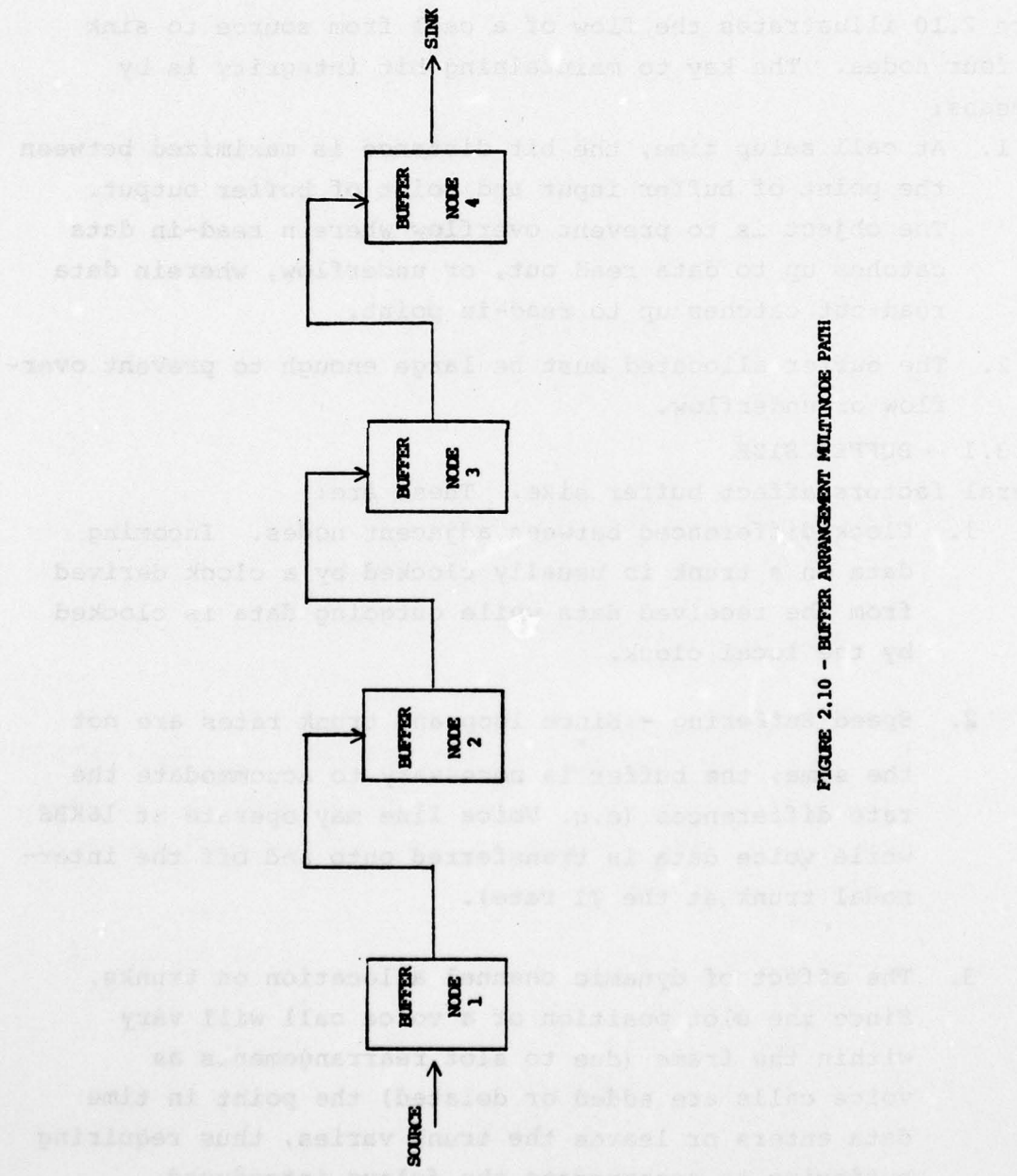


FIGURE 2.10 - BUFFER ARRANGEMENT MULTINODE PATH

Let us briefly examine the impact of each factor.

2.3.3.2 CLOCK DIFFERENTIAL

Some technique will be employed to keep the average clock differential between all nodal clocks very small. Present technology/system techniques will permit holding the maximum average daily clock differential to between 1 part in 10^8 and 1 part in 10^{12} .

The clock differential will cause filling up or emptying of a buffer, since input data rate to the buffer differs from output data rate from the buffer. Sufficient buffer must be incorporated to prevent overlap of readout and readin locations. The recommended approach is to synchronize readin and readout location of a call buffer at the call setup time. Since a synchronous data call should not extend beyond 24 hours, provision for 24 hours clock differential buffering is recommended. For typical voice rates and clock accuracies very few buffer bits are needed to accommodate this factor

2.3.3.3 SPEED BUFFERING

Entry onto or off trunks is a burst form of transmission, while voice data on loops is continuous. Obviously at least one buffer is needed to accommodate the burst data. The length of the burst is equal to the voice rate times the frame period.

$$L = V_r \times F_p$$

If the voice rate is 16 KBS and the frame period is 10 milliseconds (.01 seconds)

$$L = 16000 \times .01 = 160 \text{ bits}$$

The above indicates that buffering of at least one frame is required to accomodate speed conversion. The total amount of buffering required will be greater due to the cumulative contributions of clock differential, speed buffering, and dynamic movement of the allocated time slot.

2.3.3.4 DYNAMIC CHANNEL ALLOCATION

Any movement of the slot within the frame due to closing in of voice calls acts very much the same as clock differential since the effect of such movement is an effective change in input or output rate. The voice call can advance from the last voice call position in the frame to the first call position (See Fig. 2.11). This movement which we will term Dynamic Channel Slip (D.C.S.) can be as large as:

$$\begin{aligned} &\text{Max Voice Limit minus - (bits) Sync Pattern length minus} \\ &\text{- Call Length (bits per frame)} \end{aligned}$$

If the frame is 15,440 bits, the max voice limit might be set at 15,260 bits to insure some bandwidth dedicated to data. If sync is 32 bits long and a 16 KBS call (160 bits per frame) is the slot in question.

$$\begin{aligned} \text{D.C.S.} &= 15,260 - 32 - 160 \\ &= 15,128 \text{ bits} \end{aligned}$$

The variable "W" shown in Fig. 2.11 is another measurement of worst case Dynamic Channel Slip. "W" indicates how close the same channel can get in successive frames. The minimum "W" measured in bits equals sync pattern bit length (32 bits) plus the minimum number of frame bits dedicated for data only. At least one 16 KBS channel will be dedicated for data to handle call signalling packets (CCIS) and very high priority data packets. Since a 16 KBS channel occupies 160 bits in the frame, minimum "W" equals 192 (160 + 32) bits. The "W" factor will be employed in later paragraphs and timing charts to develop specific buffer impact due to dynamic channel slip.

2.3.3.5 BUFFER AND SYNC REQUIREMENTS

Buffer requirements must account for each of the factors previously discussed and are different for each type call local, incoming, outgoing or tandem.

Local calls will employ a common clock, have no speed conversion, and will not employ dynamic trunks. Therefore, only a single byte of storage is adequate to buffer a local call.

Incoming and outgoing calls involve a single dynamic trunk and does involve trunk to loop speed conversion

Tandem Calls utilize two dynamic trunks at the node. Although speed conversion is not required, input and outputs are non-simultaneous bursts, therefore, the individual burst per frame must be buffered. Furthermore, the clock differential between the received and retransmitted data must be compensated.

The size of the buffer, the point in time where the channel is activated and the relative start positions of input and output are the design parameters to be developed. These parameters are selected so as to maintain bit integrity, and minimize bugger and delay requirements.

2.3.3.6 BUFFER AND CHANNEL SYNCHRONIZATION RECOMMENDATIONS
Table 2.5 indicates the proposed parameters. Table 2.5 assumes that bit integrity must be maintained for 24 hours following call initialization and assumes that worst case clock differential will never cause more than 200 bits of frame slip between input and output over a 24 hour period. The maximum permissible clock differential cannot average more than 1.5×10^{-9} over the day

2.3.3.7 TIMING ANALYSIS

The diagrams shown in Figures 2.12, 2.13 and 2.14 are arrived at after some trial and error and are developed to assure minimum buffer size and minimum delay while maintaining bit integrity. All timing charts labeled A represents the possible extremes in the trunk timing. Timing Chart B in Figures 2.12a and 2.13a presents the proposed initial channel timing relative to the trunk.

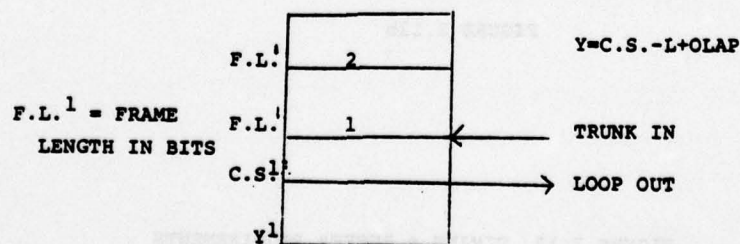
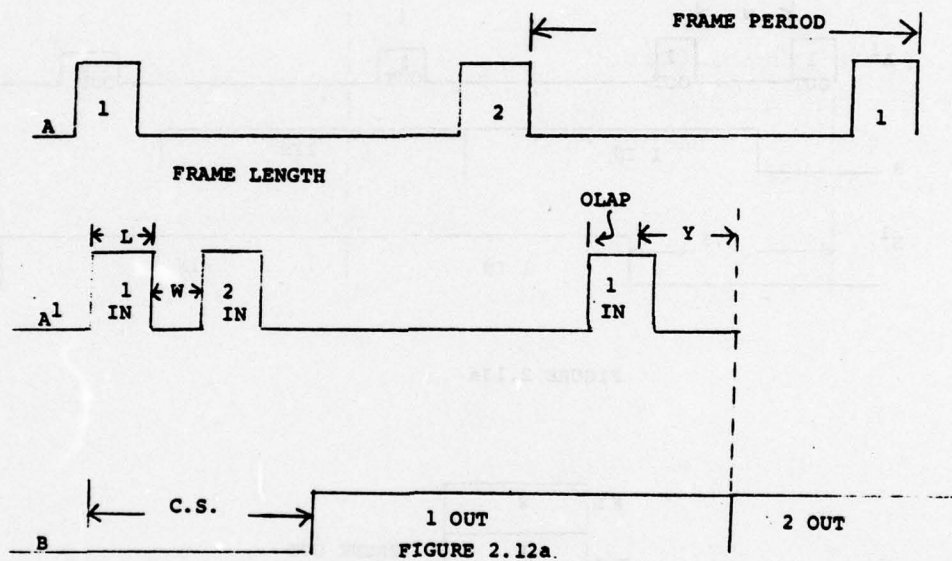


FIGURE 2.12 TIMING AND BUFFER REQUIREMENTS
INCOMING TRUNK/OUTGOING LOOP

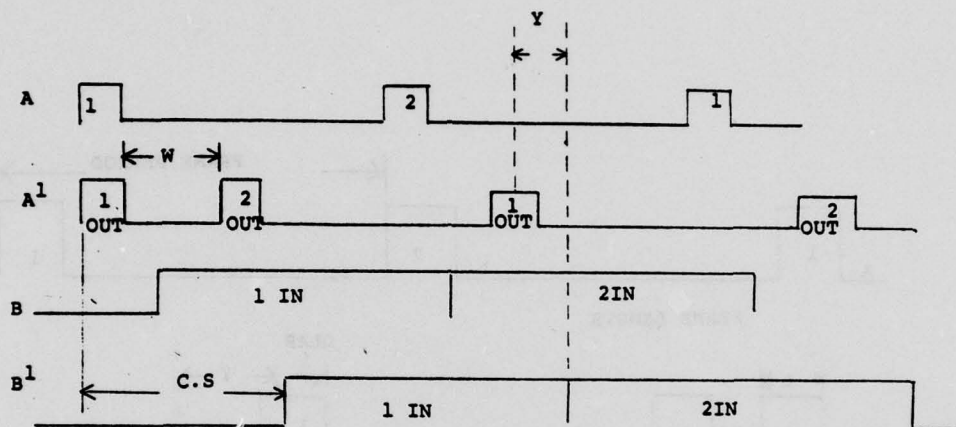


FIGURE 2.13a

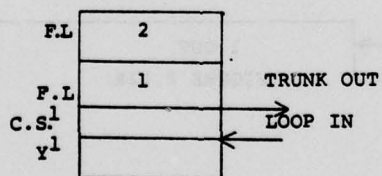


FIGURE 2.13b

FIGURE 2.13 TIMING & BUFFER REQUIREMENTS
OUTGOING TRUNK LOG/INCOMING LOOP

The offset C.S. (clock slip) is introduced to prevent buffer overflow (or underflow) in the case of worst case clock slip. Figures 2.13b and 2.13c introduce the additional buffer C.S. to compensate for offset C.S. Both C.S. and A. represent the same time delay; however, C.S. represents buffer size accumulated at the digital voice rate, while C.S. represents the higher bit buffer count representative of this buffer at the frame rate. Normally C.S. will be only a few bits in length, since the channel rate is normally much smaller than the frame rate.

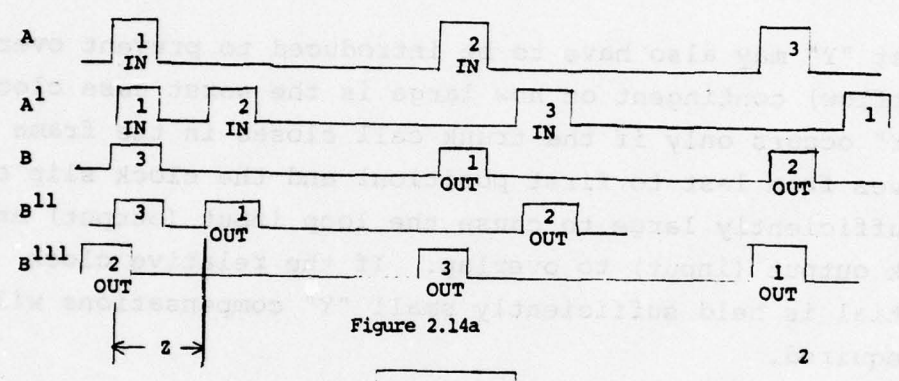


Figure 2.14a

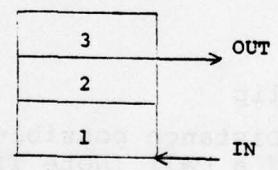


Figure 2.14b

Figure 2.14 Timing and Buffer Requirements
Tandem Call

The offset C.S. (clock slip is introduced to prevent buffer overflow (or underflow)) in the case of worst case clock slip. Figures 2.12b and 2.13b introduce the additional buffering CS^1 to compensate for offset C.S. Both C.S. and CS^1 represent the same time delay; however, CS^1 represents buffer bits accumulated at the digital voice rate, while C.S. represents the higher bit buffer count representative of bits buffered at the Trunk rate. Normally CS^1 will be only a few bits in length, since the channel rate is normally much smaller than the trunk rate.

The offset "Y" may also have to be introduced to prevent overflow (or underflow) contingent on how large is the worst case clock slip. "Y" occurs only if the trunk call closes in the frame (i.e. moves from last to first position) and the clock slip offset is sufficiently large to cause the loop input (output) and the trunk output (input) to overlap. If the relative clock differential is held sufficiently small "Y" compensations will not be required.

$$Y = C.S. - W - L - OLAP$$

C.S. = Clock Slip

W = Closest Distance possible between successive frames of a Call (Note Timing Chart A¹)

L = Slot Width

OLAP is the maximum overlap allowable of the trunk slot before input and output bits overlap.

$$OLAP = L \left(\frac{V_R}{V_T} \right) Y$$

All quantities can be expressed in terms of either time or bits referenced to the trunk rate.

The offset "Y" disappears if C.S. is small enough to cause Y to be ≤ 0 .

AD-A049 619

RCA CORP CAMDEN N J

F/G 17/2

STUDIES FOR DEVELOPMENT OF A UNIFIED NODE EMPLOYING DYNAMIC CHA--ETC(U)

DEC 77 B PATRUSKY, K BODZIOCH, R CHAN

F30602-75-C-0109

RADC-TR-77-380

NL

UNCLASSIFIED

2 OF 5
AD
A049619



Figure 2.12b and 2.13b illustrate the additional buffering Y^1 to introduce the offset Y . Y represents buffer bits to be filled or emptied at the voice rate to compensate for overlap Y . Normally Y^1 will be only a few bits in length since the voice rate is normally smaller than the trunk rate, and is very likely to be 0 if clock slip is reasonably well controlled. In general, the buffer required for incoming and outgoing links is slightly larger than two frames worth of buffering. For example, a 16 KBS call requiring 160 bits per frame would require slightly more than 320 bits of buffering. C.S.¹ and Y^1 buffers can be expanded in size to round off buffers to an integer number of 8 bit bytes.

Figure 2.14 illustrates the recommendations for tandem links. If three frames of buffering are employed, and input and output slots are started at relative positions shown in Figure 2.14b (input is first activated, then output and only on first byte of the slot) overlap will not occur unless the clock slip exceed value "Z":

$$\text{where } Z = L + W$$

Since the smallest value of L is 8 bits the smallest value of

$$A = W + 8$$

$$W = \text{Sync} + (\text{Frame Length} - \text{Max Voice})$$

$$\text{Since } \text{Sync} = 32 \text{ bits}$$

$$\text{and minimum value } (\text{Frame Length} - \text{Max Voice})$$

$$= 160 \text{ bits (16 KBS channel in 15,440 bit}$$

$$\text{Frame reserved for CCIS and other data)}$$

$$\text{Minimum } Z = 200 \text{ bits (T-1 rate)}$$

$$Z = .13 \text{ MS}$$

A maximum clock differential of 1.5×10^{-9} averaged over 24 hours is acceptable. If slippage greater than "Z" is possible, five buffers would be required and the average delay would then increase from 1.5 frames to 2.5 frames. It is recommended that clock differential be held to the recommended limit (1.5×10^{-9} or less).

2.3.4 INDEPENDENT TIMING SOURCES

It is recognized that the Defense Communication Agency (DCA) may lease much of its transmission plant from independently timed sources. It therefore may not be possible to avoid occasional loss or gain of a data bit on incoming trunks. This occurrence should be relatively infrequent on access lines since rates are relatively low. Furthermore, loss of bit integrity on access lines can only perturbate a single call. On T-1 internodal trunks the problem is more severe since many calls are affected and since the higher data rate would result in a proportionally increased rate of bit slippage.

Two alternative solutions are possible:

A synchronizing buffer could be added at the node send end of the interface to the leased carrier Data Channel Equipment (DCE). This buffer would readout bits with timing derived from the input link. The buffer would be sufficiently long that reset would not be required very often.

The second alternative would be to modify the sync detection logic to examine sync over a 3 bit window, wherein the current sync position is the center of the window. If sync is detected one bit offset from its previous position, the newly detected sync would establish the new frame sync location and a new center for the detection window. Since the sync pattern chosen will have a good autocorrelation characteristic, it is highly improbable that sync would be detected one bit offset from its true position, unless the carrier had dropped or gained a bit. Furthermore, even if the improbable occurred the sync would be detected and properly recentered in the following frame.

The second alternative (modified sync detection logic) is the recommended alternative because:

- a) The carrier network may not guarantee the desired level of bit integrity even if carrier timing is employed for sending data from the unified node.

- b) The use of the sync detection approach avoids the occasional loss of bit integrity incurred by the required buffer reset of approach 1.

2.4 IMPACT OF DYNAMIC CHANNEL ALLOCATION ON SYSTEM PARAMETERS

The purpose of this section is to analyze the technical impact of dynamic channel allocation on various system aspects as:

- . Priority/Pre-emption
- . Security
- . Routing
- . Call Setup and Breakdown
- . Service Features (conferencing, etc.)
- . Tech Control
- . Net Management/Control

Each aspect will be individually discussed.

2.4.1 PRIORITY/PREEMPTION

The only situation where voice priority/preemption might be affected by data queues is the case where "Z", "Y" or "W" precedence messages are being excessively delayed due to the existence of voice saturation at a node. In this case, it may prove prudent to permit preemption of the lowest priority Class I calls so as to expand the trunk bandwidth available for Class II data. It is recommended that the node software permit voice preemption by high precedence messages, however, such preemption should only be permitted where priority queues becomes excessively large. The basic nodal design should include sufficient dedicated data bandwidth on trunks, such that voice preemption due to data queue buildup will be a rare occurrence (.e.g. less than .1% of voice calls preempted to data queue size).

It is recommended that packet data be divided into three priorities as follows:

- . Priority I - CCIS messages, Net Control Messages and Highest Precedence data traffic (Z, Y or W).

- . Priority II - Interactive Data Traffic.
- . Priority III - Bulk and Narrative Record traffic
(other than highest precedence traffic).

Dedicated Data Channel capacity required will be adequate to accommodate all priority I traffic and sufficient priority II traffic to assure average end to end delay times in the order of one second for interactive traffic.

Since the transmission time of individual packets will be very short (i.e., 5-50 milliseconds range); preemption of "committed" packets is not recommended. A packet in the process of being transmitted should not be interrupted. However, when new packets are queued for transmit, they should be on a FIFO basis by priority. Priority I packets ready for transmission precede priority II and priority III is sent only when there is no priority I or priority II backlog. Furthermore, via use of dedicated channels and voice preemption, priority I and II backlog queues shall be rigidly kept within allowable queue size limits.

Orderly preemption of transactions progress should be permitted only in the case where an output line is busy at the time a high precedence transaction destined for the line appears at the node. The destination host or terminal controller should have the option to cancel the lower precedence transaction or place the lower precedence transaction in hold until the higher precedence transaction completes.

2.4.1.1 PRIORITY OF CCIS TRAFFIC

CCIS messages and control messages are placed in the priority I category because:

- 1) Many of these type messages serve to free buffers, transmission bandwidth and in various other ways serve to reduce node and network congestion. It is essential that these messages are not blocked or excessively delayed.
- 2) These messages represent a very small percentage of the total data traffic volume and, therefore, can be placed in priority I without significantly impacting message delays.

- 3) Call connection times are minimized.

In general, Class I calls will be preempted only by other Class I traffic, except in rare circumstances where Class II priority I data queues become excessive.

2.4.2 SECURITY

The key consideration when dealing with secure voice and data is violation of security, wherein access of secure voice or data is offered to an unauthorized party.

2.4.2.1 CLASS II DATA SECURITY

Let us first examine Class II data. In the case of Class II data, dynamic channel allocation has negligible impact as regards to security. Each packet is a self contained entity which is completely checked at the destination node prior to delivery. At the destination node, each packet is error checked (CRC code is verified), the destination address is verified, and the security level of the packet, which is transmitted along with each packet, is matched against the allowable security level of the destination line. Thus, for Class II data, intermediate transmission errors do not pose a security problem. There is always an insignificant probability that a combination of random transmission errors will cause a packet to be misdelivered, however this possibility will exist in any packet network and is not attributable to dynamic channel allocation.

2.4.2.2 CLASS I TRAFFIC SECURITY

Now let us consider Class I traffic. If "end to end" encryption is employed, the probability of a security violation is not a consideration. Even if a call path is misdirected, the unauthorized receiver will not have the matching code key to decode the information. Furthermore, since call establishment and crypto key select coordination is accomplished employing Class II packets, call set-up errors are unlikely since Class II packets are error protected.

2.4.2.2.1 LINK ENCRYPTION OF CLASS I TRAFFIC

If only link encryption is employed, then the potential exists for routing the words of a secure call sender to an unauthorized listener. Link encryption is never as "secure" as "end to end" encryption because the voice data is "red" in each switch node traversed. One way a call could be misdirected is if list frame changes initiated at one end of a link are not correspondingly changed at the opposite end of the link.

As is discussed in Section 3.1, a scheme of acknowledgements and handshaking can be employed which is practically foolproof, that is, the probability of mismatched lists can be made negligibly small, and furthermore, mismatches that occur in one frame can be corrected in the succeeding frame. Thus, even if false routing occurs for one frame, the frame time period is so small (i.e., ten milliseconds) that no intelligible output is sent to the unauthorized party. Actually, the problem of dynamic lists is less serious than the problem of trunk group sync loss. If sync is lost on the multiplexed trunk, traffic will be misdirected until the sync loss is detected. This can take several frames, perhaps as many as ten frames, in which case 100 milliseconds of the voice conversation might be misdirected. After sync loss is detected and prior to resync, the node must stop transmitting on the trunk so as not to perpetuate incorrect routing of the voice data.

For the case of link encryption of Class I data, the problems associated with security is the same type of problem encountered with any time division system, which is the proper locating and routing of slots of data. To a large degree, the security achieved is a function of the particular implementation and its safeguards against improper location or routing of time slots. The problem is, therefore, not unique to dynamic channel allocation, although dynamic movement of time slots does complicate the problem.

2.3.2.2.2 FUTURE CONSIDERATIONS

Future attempts to incorporate Time Assigned Speech Interleaving (TASI) or Time Assigned Data Interleaving (TADI) techniques would complicate the security problem. Attempts to translate link encrypted voice to a different speed in order to permit encrypted conversations between dissimilar phone terminals would also complicate the problem. This does not mean that satisfactory implementations are not possible, but rather that cost impact is considerable to add these capabilities. If "end to end" encryption is employed, the security requirements are most satisfactorily achieved, but TASI, TADI, or speed translation cannot be employed.

2.4.3 ROUTING

It is likely that the integrated network will have a profound effect on the approach to routing of Class I calls. First of all, when a choice of several output links is possible for the voice path, the decision which path to be used should be influenced by the degree of occupancy of each trunk. However, the occupancy of a given trunk will be a function not only of voice channel occupancy, but also a function of the amount of Class II data backlogged for the trunk. The data queue size and the current available data bandwidth on a trunk determine the waiting time for the last packet on the data queue. It is suggested that thresholds be established for these waiting times such that the trunk becomes unavailable for low priority Class I calls when these thresholds are exceeded. It is also desirable that a warning threshold be established which would be used in the routing algorithm to force choice of alternate trunks (when available) for new call requests. The warning threshold would be lower than the blocking threshold. Calls which cannot find alternate routes would be routed via the congested trunk (even though the "warning" threshold is exceeded) as long as the "blocking" threshold is not exceeded.

The other factor which will affect routing is the ease and rapidity with which CCIS packets can be circulated between nodes.

Rather than employing deterministic routing it is both feasible and recommended that a combination of adaptive and deterministic techniques be employed. An approach similar to the ARPA concept can be employed wherein routing tables are dynamically varied at each node on a continuously updated basis. Based on reports from neighboring nodes, each node can have a prior network knowledge of path availability, node loading, and trunk occupancy. The unified node differs from ARPA in that both voice and data must be routed. Data routing criteria can be similar to ARPA based on choosing minimum delay biased by number of path hops. Voice routing criteria would be based on choosing minimally occupied trunks biased by number of path hops, and by the size of data backlogged for these trunks. Additional study will be necessary to develop detailed routing concepts and algorithms for a network made up of unified nodes connected by multiplexed trunk groups which handle data and voice.

2.4.4 CALL SETUP AND BREAKDOWN

The call setup and breakdown procedure for Class I traffic will differ somewhat from existing circuit switches. As discussed in Section 3, some basic procedural changes recommended are:

- 1) Use of packets (instead of an out of band channel) to transmit CCIS messages.
- 2) Actual addition and deletion of a call to a trunk involves a special procedure wherein the entire trunk group frame format is altered in a manner such that both ends of the link change frames simultaneously.

2.4.4.1 CALL SETUP

The most significant change, however involves a recommendation which is directly attributable to the dynamic channel allocation concept. Since channels reserved for voice can be used to transmit data when voice is not being transmitted, it is desirable to delay committing trunk bandwidth to voice traffic until required.

Actual commitment is not required until the destination phone receiver goes off hook. Since, on the average, between 5 to 15 seconds will elapse from the time the first node receives the dialed digits until the connection is effected, it is desirable to use this period for data transmission. Thus, although the connectively path is agreed upon via CCIS packets only the final "call answer" packet should initiate the frame changes in each link of the call path and commit the trunk channel to the voice traffic. Assuming an average of 10 seconds extra data traffic can be sent per call (300 seconds average duration) the network is able to offer an additional average available bandwidth for data which is approximately 3% of the voice bandwidth. This can represent a significant cost savings in transmission cost. A slight drawback is that the connection is not established when the receiver goes off hook, but must await the final linkup. It is anticipated that worst case hookup time can be held within two seconds even if the connection path includes many hops and a satellite link. This type of delay should be acceptable considering the cost benefit attained. For most calls, the delay to connect will be very short (less than one half second). If this approach is implemented, no connection is available during the ringing interval, therefore "the ringback tone" cannot be generated at the destination node, but must be generated at the source node (following receipt of a "call complete" control message packet from the destination node). This is the recommended approach.

The other recommended change in call setup was alluded to in the paragraph on routing. The initial "call initiate" message packet can "seek out" the destination node. Each node in the network looks up its own dynamic routing table to determine what link to forward the "call initiate". In those instances where a node is reached with "all trunks busy" (not likely to occur very often since each node carries relatively up to date routing guidelines) the source node can then attempt to reach the destination via conventional deterministic routing technique.

Two types of calls request packets are required. One type is routed from node to node via the local node routing tables. The final route is established once the destination node is reached. The other type of call request is deterministically routed along a path predesignated by the source node. This call request is sent only if an "all trunk busy" response is received in response to the first non-deterministic call request.

2.4.4.2 CALL BREAKDOWN

Calls are disestablished when one of the parties goes on hook or when the call is preempted. Call release packets are sent from either end and initiate the release of associated trunk bandwidth on each link in the call path. The only difference from conventional call release is that other calls on the trunk are shifted such that the released call slot is collapsed and the Class II region of the frame enlarged. This "collapsing" slot rearrangement approach is recommended. Since call slot sizes will vary with call bandwidth, it would be difficult to utilize random free slots in the frame. Closing in the voice calls in the frame whenever a call is released consolidates all the available bandwidth, making it readily usable for transmission of data or for establishment of new voice calls.

2.4.4.3 PREEMPTION

2.4.4.3.1 DESTINATION CALL PREEMPTION

Since many of the voice calls will employ end to end encryption, it will not be possible to connect a preemptor to his called party without resynchronization. The preferable approach for preemption appears to be to force the intended receiver to hang up so that the preempting call can be established as a normal call setup. The preemption process would operate as follows:

- 1) The preempting source node sends a high priority "call initiate" to the destination node.

- 2) If the destination line is busy with a lower priority call, the destination node places a preempt tone on the receiving line and simultaneously sends a "preempt release" packet on the return path to the source node.
- 3) The "preempt release" packet generated by the destination node and sent back to the source node initiates breakdown of the call at all intermediate nodes and causes the source node to provide a preempt warning tone to the local subscriber, followed by disconnect.
- 4) When the called party hangs up, the destination node initiates ringing, and returns a "call complete" packet to the preempting node, thus initiating ringback.
- 5) A new call is established in the normal manner. (Receiver goes off hook and the "call answer" packet is sent over the connect path).

2.4.4.3.2 TRUNK CHANNEL PREEMPTION

It is possible that a high priority call will have to preempt a trunk channel which is handling an unrelated lower priority call. This should not be a common occurrence. To avoid additional complexity, it is recommended that the trunk call preempted be required to have the same or higher digital data rate (otherwise more than one call would have to be preempted). The call preempted would be the oldest call on the trunk of lowest priority. Preemption need not be initiated till the "call answer" packet is received at the intermediate node, which will initiate preemption. The intermediate node passes the "call answer" packet back to the originating node (of the new call), sends preempt release packets to the source and the destination nodes of the call being preempted (along the call path.) Following ACK of the preempt release from the adjacent nodes, the intermediate node preempts the trunk and relays the call answer packet back towards the preempt source.

2.4.5 SERVICE FEATURES

User service features are not impacted by dynamic channel allocation. However, secure digital voice networks and packet switch networks do differ from conventional circuit and message switches and introduce certain constraints on potential service features. Let us first examine Class I communications.

2.4.5.1 CLASS I SERVICE FEATURES

Those service features involving establishment of a point to point call offer no special difficulties for the unified node implementation. These include such features as:

- Precedence privileges
- Abbreviated Dialing
- Compressed Dialing
- Call Forwarding
- Group Line Hunt
- Camp On
- Hot Line (Direct Access)
- Automatic Ringback
- Conference Privileges

The previous listing is not to be construed as recommended service features, but rather as a list of options whose costs are consistent with costs of providing these options in conventional circuit switches.

Those features involving third party connections are somewhat more complex to implement especially in the case of secure digital voice traffic. Features in this category are:

- Attendant/Recall Operator Function
- Conferencing

2.4.5.1.1 ATTENDANT/RECALL OPERATOR FUNCTION

If a digital or secure conversation is ongoing and one of the parties desires to reach a system operator without disconnecting

the ongoing conversation a means must be provided to flag the operator and establish a non secure link to the operator. Unlike conventional systems, the most practical implementation will not permit the third party to monitor the conversation between the operator and the other party requesting operator action. Furthermore, a special operator request key on the terminal is more practical than attempts to use the conventional on hook/off hook tapping to gain operator access. This request key when held down would also operate the crypto bypass. In the event the destination party requests a call transfer, special software will also be needed to provide the correct crypto key to the new party and to initiate crypto resynchronization. Thus, operator recall impacts network and terminal equipments.

2.4.5.1.2 CONFERENCING

At the present time no practical conferencing technique has been developed which handles encrypted voice. One practical approach to conferencing secure voice is to employ decryption/encryption units. Each end to end path then consists of two encrypted links; one from the terminal to the conference input circuit and then from the conference output to a receive terminal. It is possibly desirable to locate all secure conference circuits at a common site such as the network control center so that necessary physical and electrical security can be concentrated at one center. If the conference circuit employs D/A conversion, the conference circuit can also be employed to accomplish connections between terminals with dissimilar phone rates. Certainly if the network must service many users which employ a wide diversity of cryptographic techniques; data rates; or analog/digital conversion techniques; the number and type of conference circuits required will grow. It is suggested that conference circuits required be developed and tested in a breadboard laboratory environment to assure good voice quality, proper connection and breakdown, and adequate security safeguards.

2.4.5.1.3 PRECEDENCE/PREEMPTION

As discussed previously, this feature poses no special problems, however, ruthless (immediate) preemption is recommended so as to reduce the time for the preemptor to obtain his connection. A preempt tone at least one half second in duration should occur prior to release of the preempted circuit to signal the impending disconnect to the connected parties.

2.4.5.2 CLASS II SERVICE FEATURES

The intent of a packet network is to establish a transparent reliable network useable for data communications. Although packets are buffered to avoid the rigid time constraints of a synchronous network, the intent is to provide for near real time data communications. No attempt will normally be made to keep a permanent record of packet data entering or leaving a node, thus, the following services will not be provided by the packet network:

- 1) Archival Storage
- 2) Message Retrieval
- 3) Intercept

Since some (or all) unified nodes may contain a conventional message switch, a limited number of users may be able to obtain these services via access to the store and forward message switch, however the traffic must then incur the added delay of being routed to and from the message switch.

Certain service features available to a message switch on a per message basis are handled by the packet switch on a per packet basis, namely:

- 1) Precedence handling
- 2) Security
- 3) Mode/Speed Conversion
- 4) Trace

The following features require separate discussion because they require unique implementations or differ substantially from their message switch counterparts:

- 1) Accountability
- 2) Multiple Addressing
- 3) Transaction Preemption
- 4) Packet Circuit Type (Virtual vs Datagram)

2.4.5.2.1 ACCOUNTABILITY

In a packet system, host computers external to the network assume a large responsibility in assuring that messages received are accounted for and accepted by designated users. The particular accountability features of the hosts is not a part of this study and even may be unique to each host. Suffice it to say that the network responsibility ends after ascertaining that each segment of a given transaction is received from the source host and delivered to the destination host. To this end, each segment received from a source host contains a sequence number. Upon successful delivery, the sequence number is checked off. The final segment is acknowledged by the destination host with a "Request for Next Message" (RFNM). A link protocol (ADCCP) provides the link segment accountability and segment retransmissions as required. The combination of link and segment interface protocol constitutes the total degree of network transaction accountability.

Segment accountability is based on link and destination node acknowledgement of segment delivery and checkoff of sequence numbers. The destination node will usually require that segments be received in sequenced order and controls segment flow such that only one segment at a time is in transit for a given transaction.

In order to reduce overhead and speed transaction delivery, it is recommended that the network also permit certain hosts to send multiple segments to each other in any sequence. In this case,

the receiving host assumes the responsibility for ordering received segments and controlling data flow, however the network is still responsible for assuring that all segments have been delivered to the Hdst.

2.4.5.2.2 MULTIPLE ADDRESSING

Since packet transactions are delivered in near real time, it is difficult for a packet network to handle multiple addressing.

The most serious difficulty would be the situation where one destination could accept delivery but the other could not. In general, the worst case delivery time dominates; that is to say that each packet of a multiaddressed message would have to await acknowledgements from all destinations prior to transmitting the succeeding packet in the transaction.

Since computer-computer and interactive traffic is generally not multiaddressed, it is recommended that initial nodes developed do not incorporate a multiaddressing capability in the packet network. However, this feature would still be available to those users that can use the store and forward service offered by the message switch.

Further study is warranted to determine the desirability and impact of incorporating a multiaddressing capability into the packet switch.

2.4.5.2.3 TRANSACTION PREEMPTION

In general, a busy hour transaction will range in delivery time from one second to about twenty minutes depending on transaction length. The question arises whether or not to permit preemption of a transaction and if preemption is permitted, how to mechanize recovery. It is readily noted that if an orderly halt can be achieved such that a known packet sequence number is delivered and the next packet of the transaction lies resident in the source

host, interrupted transactions can be resumed without requiring transaction restart.

RCA has implemented such an orderly halt procedure in the ADPT testbed and it has proven feasible. Transaction preemption is recommended if the source or destination node requires buffers to accommodate new higher priority transactions and buffer availability has fallen below some designated threshold. Such preemption should be orderly permitting resumption at the point of interruption. Sufficient buffers should be provided in the node such that transaction preemption occurrences are few, and recovery relatively rapid, or transaction delivery delays will be adversely impacted.

Additionally, preemption will be necessary if the addressed destination line is fully occupied and one or more of the existing transactions is lower in precedence than the new data transactions.

2.4.5.2.4 PACKET CIRCUIT TYPE

Several types of packet circuits can be offered by the network, namely:

- Datagram Service

- One Way Logical Circuit

- Two Way Logical Circuit

The datagram is a single segment which is sent through the network to a destination and does not require any automatic acknowledgement back to the source node. Accountability lies totally in the hosting computers. This type service does not have sufficient accountability for a military network and should not be offered.

The one way logical circuit was implemented in the ADPT testbed. The first packet of the transaction establishes a one way data path (if accepted). The last packet of the transaction disestablishes the virtual circuit. Flow control messages are returned

to control data flow and acknowledge received packets, however, no data return path is provided.

The two way logical circuit was not implemented but is not significantly different in implementation from the one way circuit. Special control messages establish and disestablish a two way data link. These messages may or may not be appended to data packets. This circuit is useful for interactive or query/response exchanges where a return data link is required since separate circuits need not be established.

The future node should permit the establishment of both one way or two way circuits. In terms of implementation, this requires that separate designators be employed in the packet header to indicate "End of Message" "End of Transmission". If a message requires an answer, the return link releases the circuit only after providing the appropriate data response. In addition, separate designators must indicate the data direction (forward or answer).

2.4.6 TECHNICAL CONTROL

The capability to monitor sync on the multiplexed trunks coupled with the ability to exchange test messages over these trunks provides for general test and checkout of nodal trunk operation. It is also important to detect those failure modes which affect channel slot boundaries on multiplexed trunks. A useful monitor is the ability to check for buffer overflow or underflow on a per channel basis, since this type of alarm will usually be triggered when channel slot boundaries are not correct. Use of the above described tests permits fairly comprehensive checkout of the internodal trunk lines employing dynamic channel allocation. Since data at each trunk end terminates in memories accessible by processors, tech control of these trunks can be completely automated.

Data access lines can be tested automatically employing the test message exchange specified as part of the ADCCP.

Phone lines can only be partially tested automatically. The chief limitation is that phone terminals are not usually designed to detect or generate test tones when on hook. Automated test of phone access lines is technically achievable but will become prohibitively expensive if not included in those programs developing the new digital phones.

2.4.7 NETWORK MANAGEMENT

The ability of nodes in the network to send and receive network control packets to and from the Network Control Center in a rapid and reliable manner simplifies the communications required for good network management. The net control center can maintain timely tables on system status, traffic loads, line and trunk failures, etc. and can, therefore control traffic distribution and maximize system "up" time.

2.5 USEFUL RANGE OF OPERATION

Dynamic Channel Allocation is an extremely useful concept. The key advantages are:

- 1) The ability to efficiently accommodate a wide range of data rates.
- 2) The ability to reduce the transmission bandwidth required in the network.
- 3) Reduction in tech control, modem and crypto costs are realized by multiplexing integral with the switch.
- 4) It is obvious that the ability to satisfy varied user requirements with a single system results in reduced acquisition, production and life cycle costs (when compared against using independent switch networks for different type user group). A specific cost trade study will be required to determine detailed cost savings.

This section addresses the practical limitations of the dynamic channel allocation concept.

2.5.1 MAXIMUM TRUNK RATE

The maximum trunk rate is practically limited by memory access speeds. The BMX implemented in the ADPT testbed indicated that as many as 7 control memory access cycles can be required to derive a single data memory address for transfer of data to or from the internodal trunks. Based on present day memory speeds a data memory with 500 nsec cycle time supported by a control memory with about 70 nsec cycle time is a realistic design limit without resorting to highly specialized ultra-high speed circuits. Although an eight bit slot size is the recommended line access rate the control memory could be used to build a full word (i.e. 32 bits in length) for transfer to and from memory. The resultant thrupt attainable is 64 MBS. Assuming that 5% of the accesses involve overhead data handling the present day thrupt limit is about 60MBS. If this thrupt was used to accommodate a single wideband full duplex trunk, the trunk rate would be 15MBS (each trunk must support data thrupt in two directions and all transfers come into or out of access lines, hence a four to one ratio). The maximum trunk rate might be further upgraded by employing shared memory since data inputs could address different data banks simultaneously and memory cycles can be overlapped however esoteric synchronization techniques may have to be employed to reduce shared memory access conflicts. At most an additional 20-40% speed gain can be attained employing shared memory in a realistic design (raising the limiting trunk rate to about 18MBS.)

It is probable that memory speeds will improve from 50-100% in the next decade permitting a proportional upgrading of the maximum trunk rate.

The trunk capacity available can be split among multiple trunks thus about 9-10 T1 rate trunks can be supported by a single

central BMX without shared memory speedup and from 12-14 T1 trunks with shared memory. This does not imply that larger nodes are not realizeable since it is possible to operate with parallel modules, however the single trunk limit in the range of 15-18 MBS still applies.

2.5.2 USER CHANNEL RATE

It should be obvious that if a single user occupies a very large proportion of the trunk bandwidth, the presence or absence of this user on the trunk can radically affect the desired grade of service on the trunk. This is not desirable. Let us take the specific case of T1 trunk. The total trunk capacity of a T1 trunk is approximately

- 192 - 8 KBS Channels or
- 96 - 16 KBS channels or
- 48 - 32 KBS channels or
- 24 - 64 KBS channels or
- 12 - 128 KBS channels or
- 6 - 256 KBS channels

Based on a blocking probability of 1% the following would be the offered loaded normalized to rate permitted for each of the above cases (obtained from Telephone Traffic Theory - Tables and Charts part 1 assuming infinite sources and full availability) for a trunk operating at the T1 rate.

RATE	OFFERRED LOAD T1 (1% BLOCKING)
8 KBS	1376 KBS
16 KBS	1330 KBS
32 KBS	1155 KBS
64 KBS	978 KBS
128 KBS	753 KBS
256 KBS	489 KBS

Based on a blocking probability of 5% the corresponding values are:

CHANNEL RATE	OFFERED LOAD T ₁ (5% BLOCKING)
--------------	---

8KBS	1521 KBS
16KBS	1458 KBS
32KBS	1360 KBS
64KBS	1216 KBS
128KBS	1018 KBS
256KBS	758 KBS

These previous numbers indicate that the allowable offered load reduces considerably if high rate users constitute a large proportion of the offered load and the effect is more pronounced as trunk blocking requirements become more stringent.

It becomes apparent that a penalty in transmission efficiency must be paid to accommodate higher data rate users and still achieve the desired blocking probability.

The allowable offered load can be approximated by a weighted average of maximum offered loads for a given traffic mix.

Thus assuming the traffic mix is:

		WTD LOAD 1%
8 KBS	20%	275 KBS
16 KBS	40%	532 KBS
32 KBS	20%	231 KBS
64 KBS	10%	98 KBS
128 KBS	5%	38 KBS
256 KBS	5%	<u>25 KBS</u>

The allowable offered load = 1199 KBS

This represents a 10% loss in the offered load, compared to the case where all calls are 16 KBS. This would approximate a corresponding loss in transmission efficiency, except that in a node employing dynamic channel allocation some of this bandwidth would be available for Class II data. Since each 256 KBS call

represents 17% of the trunk capacity, it is readily apparent that no more than one call of that rate is tolerable on a T-1trunk since even the one call would be much higher than the average traffic load offered at that rate.

It is recommended that the maximum call rate to be handled by a T-1trunk be limited to about 200 KBS and that only one 200 KBS call be permitted on the same trunk simultaneously. It is further recommended that the percentage of average traffic contributed by users operating at rates higher than 64 KBS do not result in a drop in composite offered load greater than 45 KBS or 3% of the T1 rate.

This may necessitate employing alternative schemes or special wideband trunks (above the T-1rate) to handle higher data rates, if the amount of such traffic justifies.

3.0 COMMON CHANNEL INTEROFFICE SIGNALLING (CCIS)

The first part of this section examines several approaches for handling CCIS signalling and has recommended an approach which relies heavily on transferring CCIS information in the form of packets. Class I call establishment and disestablishment procedures are traded.

The second part of this section develops recommended formats for CCIS packets and for Class II data packet headers.

3.1 CCIS APPROACH STUDIES

The network nodes must communicate with each other to exchange control messages required for establishing, and releasing voice and other Class I calls. In addition, other control information must be exchanged between nodes. In the proposed system, nodes are interconnected by trunk groups. Each trunk group will multiplex (time slot) synchronous data (voice etc.) and asynchronous data (packets). The CCIS traffic will be transmitted over each trunk group.

3.1.1 CCIS TRANSMISSION APPROACHES

There are several ways in which CCIS information can be transferred, namely:

- In Band
- Dedicated Out of Band
- Asynchronous packets

3.1.1.1 IN BAND APPROACH

In this approach the bandwidth on the trunk normally used for voice transmission is used for call setup and breakdown. This approach is rejected for the dynamic channel allocation system because:

- a) Since voice bandwidth during call setup can be gainfully used for data transmission, any requirement for early occupancy of a voice channel for "In Band" signalling

would defeat the key advantage of "In Band" signalling which is bandwidth conservation.

- b) It is more complex (costly) to extract signalling information which can be randomly present in non consecutive time slots of a trunk group than to extract same from an out of band channel.
- c) Since data does not occupy dedicated channel bandwidth, the "In Band" approach can only apply to CCIS traffic associated with voice calls. A unified approach which accommodates both voice and data CCIS traffic is preferred.

3.1.1.2 DEDICATED OUT OF BAND APPROACH

The dedicated "Out of Band" (OOB) approach would allocate one or several time slots in the trunk group frame for transmission of signalling and supervision data.

The key advantages of this approach are:

- a) It is simpler to implement than the In Band approach.
- b) The capacity available for transfer of CCIS data is well defined.
- c) All control messages can be handled with a common approach.

The key disadvantages are:

- a) The OOB channel represents transmission overhead which is not always utilized.
- b) CCIS data is always delayed part of a transmission frame, even when trunk group occupancy is low.
- c) The OOB channel must be continuously monitored for presence of CCIS traffic presenting a continuous and fairly heavy processing load.

3.1.1.3 USE OF ASYNCHRONOUS CCIS PACKETS

For a node which integrates voice call handling and data packets, CCIS information can be transmitted as data packets. This is the

recommended approach for the following reasons:

- a) It is the easiest approach to implement because the mechanization of packet transfer is already required for transfer of Class II data.
- b) It is an efficient approach (from the transmission viewpoint) since CCIS packets can vary in length according to the amount of supervisory data to be transferred, since bandwidth is occupied only when CCIS messages are generated, and since more than one CCIS packet can be transmitted per frame.
- c) The error control required for CCIS messages is inherent in the packet protocol, (i.e. no additional features are required).
- d) The availability of unused voice bandwidth could provide added bandwidth to speed CCIS information transfer, especially during periods where voice occupancy is not at the blocking level.
- e) The receiver node processor need not interrupt every frame to look for call signalling, thus reducing interrupt overhead processing. This is because we treat CCIS packets as we do data packets.

3.1.2. CCIS PACKET BLOCKING AND DELAY PROBLEMS

A problem does exist when all voice channels in the trunk group are occupied. If the voice traffic were to be allowed to fully occupy the trunk group bandwidth, a condition could occur where no CCIS traffic can be transferred. In fact, no means would be available to send "Call Release" messages to terminate the voice calls occupying the trunk group. It is therefore necessary to prohibit voice from fully occupying the trunk group and dedicating a portion of the available bandwidth for data. One may then ask how does this differ from employing a dedicated "out of band" channel. The answer is that bandwidth reserved for data channels can be used for transfer of normal data packets as well as CCIS

packets. The next question then arises as to the possibility the CCIS packets may have to wait in long data queues to be transmitted, thus resulting in undesirable delays (i.e. in excess of a second).

The solution is to treat CCIS packets as highest priority packets, thus ensuring their prompt delivery. Since a packet priority system for serving output queues is needed for handling other type data packets, the impact of priority handling of CCIS data is small.

3.1.3 FRAME CHANGING

As voice calls are added and deleted, the composition of the trunk group frame must change. In order to maintain existing calls and prevent insertion of garble bits, it is vital to change frames at "send" and "receive" ends of the link simultaneously. One approach to achieving "frame change synchronization" is to alter the frame immediately following receipt of a CCIS packet acknowledging a "call answer" or "call release" request. This type approach has several major shortcomings:

- a) There is no absolute way to guarantee that the processing system will be able to get around to processing the CCIS acknowledgement packet within one frame time.
- b) It is extremely difficult to synchronize "send" and "receive" ends during frame change.
- c) There is no verification that frame change has occurred.

For the stated reasons it is recommended that a small portion of each frame be dedicated for frame change commands. The following type commands are required:

FRAME A - USE FRAME A

FRAME B - USE FRAME B

The commands are sent in a fixed position in the frame and are orthogonal to each other, such that a best choice decision can be made (which command is sent) even in a very high error environment.

As indicated in Section 2, two complementary sync patterns can serve the dual function of locating frame sync and controlling frame changes.* This approach reduces the generation and detection hardware and the overhead for accommodating frame change commands.

After a frame change is agreed upon by packet signalling, the receiver node readies the standby frame list for frame change. Standby list A is ready if the on-line list is the B list. The receiver changes to frame list A on the receive leg upon detecting the frame A command. The return leg is changed in the same manner.

3.1.3.1 PROTECTION AGAINST NON-COORDINATED FRAME LISTS

Before considering the problem of frame initialization and assuming that frame lists are coordinated, it is necessary to insure that one node does not switch lists without the other following. Two safeguards are recommended which prevent this from occurring (if the equipments are operating without malfunctions).

- a) The frame list change must occur in both directions (since voice trunk occupancy is full duplex) therefore a frame change generated in one direction (source to sink) shall elicit a frame change in the return direction. Thus the source node monitors for the altered frame command on the return leg. If frame change does not complete within an allotted time, the source node initiates recovery procedures.
- b) The frame command can be missed, however since the frame commands (A and B) are switched each time a node changes frame the "missed" command will be detected in the succeeding frame. For example, assume the receive node is receiving frames in the "A" state and waiting for a

* It is noted that rules for detection of sync and frame change are not the same.

"B" command. Even if the first "B" command is missed, subsequent frames received contain the "B" command. The receive node finally recognizes the "B" command frame and processes the receive frame with the "B" list. At the next start point of the return frame to be sent by the receive node, the frame command "B" is sent and frame list "B" is used to generate the return frame. Both legs now operate in the "B" state and remain in this state until the next occurrence of a frame change.

3.1.3.2 CONFLICT RESOLUTION

It is necessary to prevent a glare condition (that is, both switches simultaneously attempt to change the trunk frame).

One approach to handling conflicts is as follows. Each node at each end of the link checks to determine if a CCIS frame change request packet has been received from the other node prior to sending such a request to the other node. If a frame change request has been received the node ACK the received frame change and holds up sending its own request. If a frame change request is received after a similar request has already been transmitted by the node, a negative acknowledgement is returned. If a negative acknowledgement is received to a frame change request and a negative ACK has also been sent both ends repeat the procedure following a short random time delay until one end is finally cleared to start the actual frame change.

An alternate approach is to designate the node at one end of a link the master and the other a slave. The master always initiates frame changes. These changes may either be started by the master independently or as a result of receiving a frame change request packet from the slave end. The master/slave approach is more straightforward and easier to implement, and is therefore recommended.

3.1.4 CONCATENATION OF CCIS MESSAGES

The recommended approach of sending CCIS messages as high priority packets provides a fast, efficient transfer mechanism for these type messages for typical links, however a problem does arise when long link delays are encountered such as will occur over satellite links. In this type of case, the "frame change" procedure takes approximately one second. If the call requests for frame change should exceed one call per second (total for both directions) requests will queue and long connect delays can be encountered. Since a total call connect time longer than two seconds is not desired, a solution must be found to speedup the frame change process.

The recommended solution is to permit queue buildup of frame change request during periods when the frame change process has not been completed. Upon completion of the previous frame change all queued requests are concatenated to form a single frame change command packet. A 2000 bit packet would permit at least eight call changes to be concatenated into a single command. The designated master end initiates all frame changes. The slave end forwards its call requests to the master end, so that the master end can execute the frame change. The concatenated packet commanding frame changes also serves as the acknowledgement that the master received and processed the slave requests. The suggested sequence is as follows:

- 1) Slave sends call requests to master as they arrive.
- 2) Master accumulates its own and the slave requests and prepares and transmits the concatenated packet.
- 3) The slave acknowledges receipt of the concatenated frame change command packet and declares to be ready for frame change. The slave also checks off its own requests that were reflected in the concatenated packet.

- 4) Master initiates frame change sequence on master slave leg.
- 5) Slave responds with frame change sequence on slave master leg.

This approach also eliminates the problem of the added delay that would occur if each end were to take turns initiating the frame change.

Other techniques considered were to transmit frame change packets in certain time windows and change the frame in succeeding time windows. However, such an approach increases the probability of lost requests and complicates the problem of keeping frames coordinated.

3.1.5 RECOMMENDED APPROACH SUMMARY

- 1) Transfer CCIS messages as packets.
- 2) Employ priority queuing on output trunks and assign CCIS packets the highest data priority.
- 3) Provide sufficient dedicated data bandwidth on the trunk to insure meeting maximum CCIS packet delay limits under conditions of maximum voice occupancy of the trunk.
- 4) Use sync portion of frame to transmit the frame change coordination commands, specifically
 - Frame A (Sync A)
 - Frame B (Sync B)
- 5) Resolve glare conflicts using the master/slave approach.
- 6) For satellite links (and perhaps for all links) incorporate the capability to incorporate several channel changes within one frame change command packet.

It should be noted that the following CCIS messages will trigger initiation of the frame change procedure:

- a) Preempt Release
- b) Normal Release
- c) Call Answer

3.2 MESSAGE FORMATS

The unified node will send a variety of signalling and supervisory messages between nodes broadly categorized as:

- a) Voice CCIS (Class I traffic)
- b) Data and Control Messages (Class II traffic)
- c) Net Management messages
- d) System Control messages

All of the above administrative traffic can take the form of packets. The CCIS packets for Class I call handling and the control messages for handling data transactions are similar functionally, however certain key differences exist. It may be desirable to initiate a data transaction by appending the "Call Initiate" request to the first data packet to avoid the added delay involved in waiting for acknowledgement prior to proceeding with data transmission. A greater impetus exists in the case of data headers to compress field lengths such as address fields since the data header has a far greater impact on transmission overhead than voice CCIS messages (since data headers occur in every packet, while voice CCIS messages occur on a per call basis (much less frequently)).

The message formats to be described will accommodate multiple networks. Each network can contain up to 80 unified nodes (DCA planning documents indicate future plans for a backbone network containing between 60-70 nodes). It should be recognized that since such formats are being developed concurrently by Autodin II contractors and commercial carriers final formats will probably be adjudicated by committee. The formats recommended in this study provide a basis for transmission overhead analysis, highlights certain problems and tradeoff considerations involved in developing complete and efficient formats, and offer a baseline recommendation.

The following documents were digested and used as source aids in developing the format recommendations.

- a) Advanced Research Projects Administration (ARPA) network interface message processor (IMP) specification dated
- b) Digital Common Channel Trunk signalling/supervision specification Appendix VI Spec. No. TT-B1-1201-0030 dated 7 April 1976 (TRI TAC).
- c) Defense Communications Agency System performance specification (Type "A") for Autodin II Phase I November 1975.
- d) CCITT recommendation X.25 (proposed)
Interface between data terminal equipment and data circuit termination equipment for terminals operating in the packet mode on public data networks.

3.2.1 CLASS I MESSAGE FORMATS

CCIS message formats for voice and dial-up circuit switched data calls, have been well defined for military switches such as the TRI-TAC switch (TTC-39). Rather than invent an entirely new set of CCIS formats, it appears desirable to adapt an existing standard, such as the formats to be employed by the TRI-TAC system.

(See 3.2.1.3)

This approach is attractive from the standpoint of protocol and software standardization, and would simplify interfacing the unified node network and elements of the TRI-TAC system.

The basic recommendation is to packetize the TTC-39 CCIS messages for Class I traffic by enveloping these formats with the prefix and postscript data required for packet link control as described in the "Advanced Data Communications Control" Procedure (ADCCP). ADCCP is a well defined link protocol now being drafted as an ANSI standard for packet data link control and error protection and is the recommended link protocol for packet communications. In addition to the packetizing requirement other changes are recommended and discussed in the next paragraph.

3.2.1.1 MODIFICATIONS TO BASIC TTC-39 FORMAT

Figures 3.1 through 3.7 are extracts from Appendix VI of specification TT-B1-1201-0030 dated-7 April 1976. The formats are reproduced for information and discussion. Detail descriptions of the message functions, and field functions and formats are to be found in the same appendix.

The following modifications to the format are recommended for use in the unified node network under study.

- 1) Each of the presented formats are to be preceded by 2 (or 3) octets for link control and succeeded by four octets comprising the Cyclic Redundancy check (CRC) code when transmitted as described by ADCCP.
- 2) In those cases where multiplexed trunk groups are used for voice or voice/data combinations the "trunk number" field in the formats depicted must be replaced by the "trunk group" number.
- 3) Although the actual "SOM" character code can remain unchanged; it is recommended that the first non ADCCP character be a "Format Identifier" which is unique to each packet class or category. Packets associated with Class I traffic would have a unique code (which can be identical to the "SOM" code).
- 4) Since packet error checking would be accomplished via the CRC check, parity bits and the parity sum check are redundant and present an unnecessary processing load. It is suggested that these bits be replaced by fixed "ones" or "zeroes" to avoid processing. Parity would have to be checked and generated only at gateways to the TTC-39.

3.2.1.2 ADDITIONAL POTENTIAL MODIFICATION

Due to the fact that each octet in the TTC-39 format contains a control and parity bit, any information field exceeding six bits is non contiguous and crosses octet boundaries.

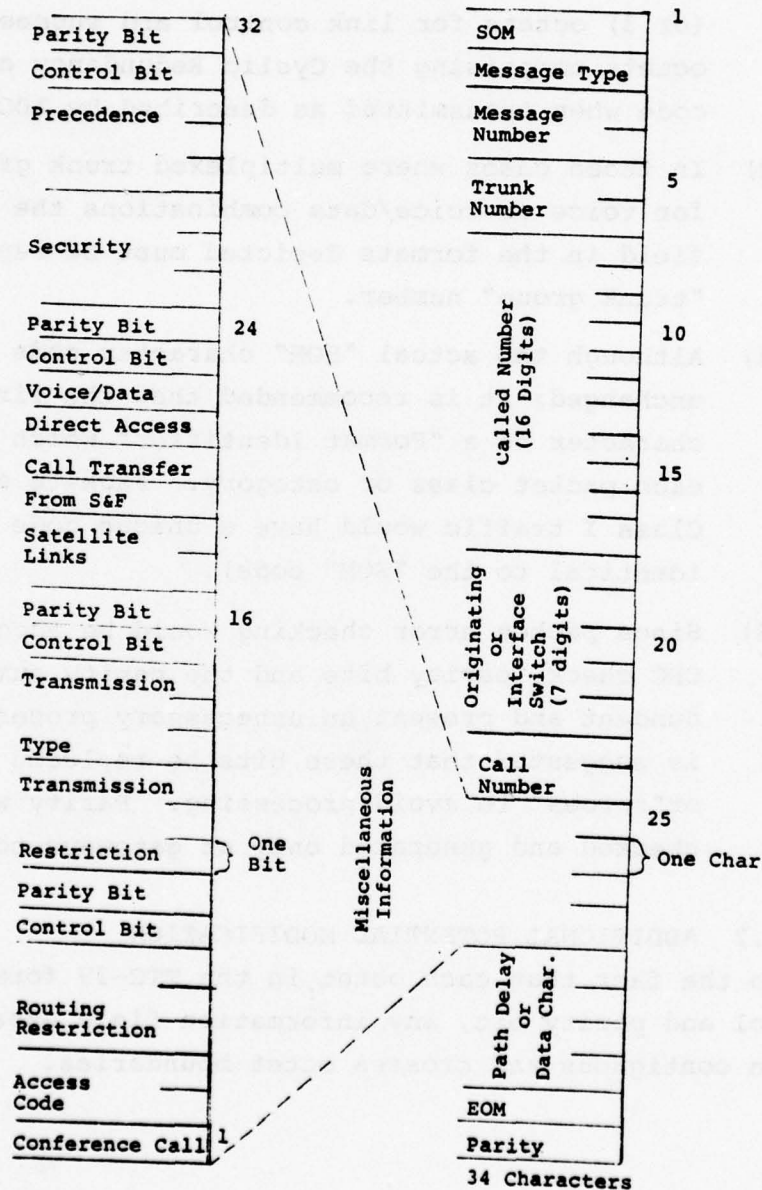


Figure 3.1 Message Format for CALL INITIATE

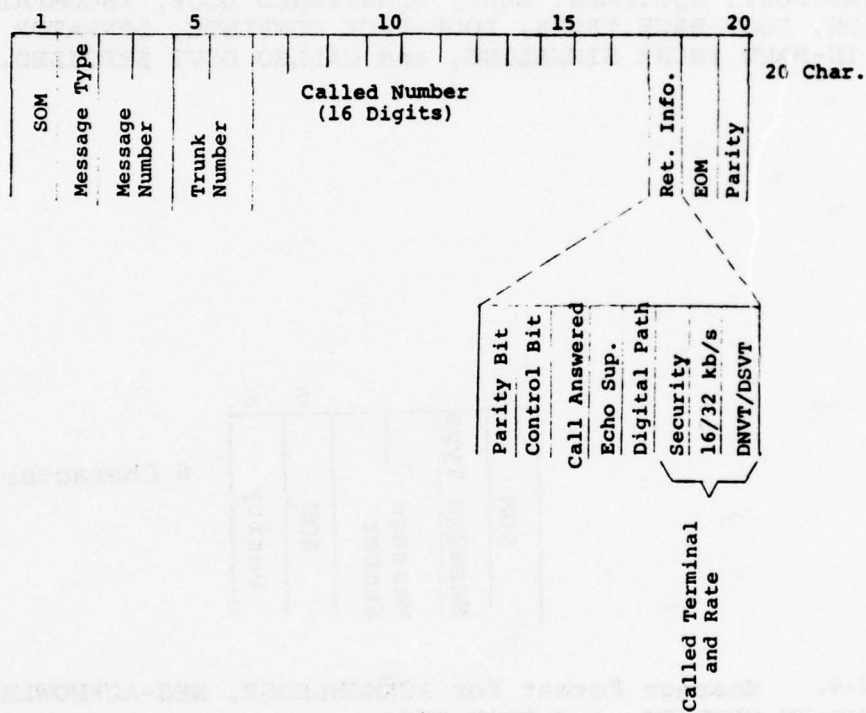


Figure 3.2 Message Format for CALL COMPLETE

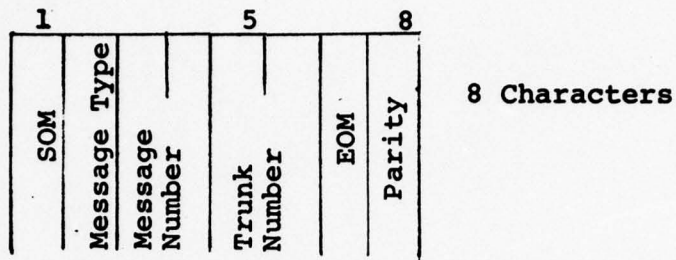


FIGURE 3.3. Message Format for CALL ANSWER, CALL RELEASE, PREEMPT RELEASE, BUSY RELEASE, CALLED PARTY UNAVAILABLE, ALL-TRUNKS-BUSY, EQUIPMENT BUSY, UNASSIGNED LOOP, INCOMPATIBLE CONNECTION, LOOP-BACK TRUNK, LOOP-BACK COMPLETE, OPERATOR RECALL, REQUEST IN-BAND TRUNK SIGNALING, and CALLED DSVT ZEROIZED.

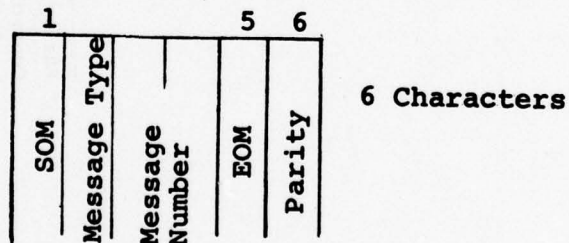


FIGURE 3.4. Message Format for ACKNOWLEDGE, NEG-ACKNOWLEDGE, GLARE, OUT-OF-SERVICE, and TEST SYNC.

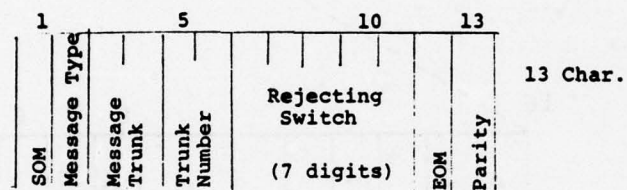


FIGURE 3.5 Message Format for INVALID ROUTE

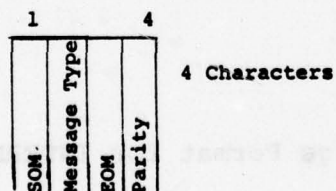


FIGURE 3.6 Message Format for SYNC A and SYNC B

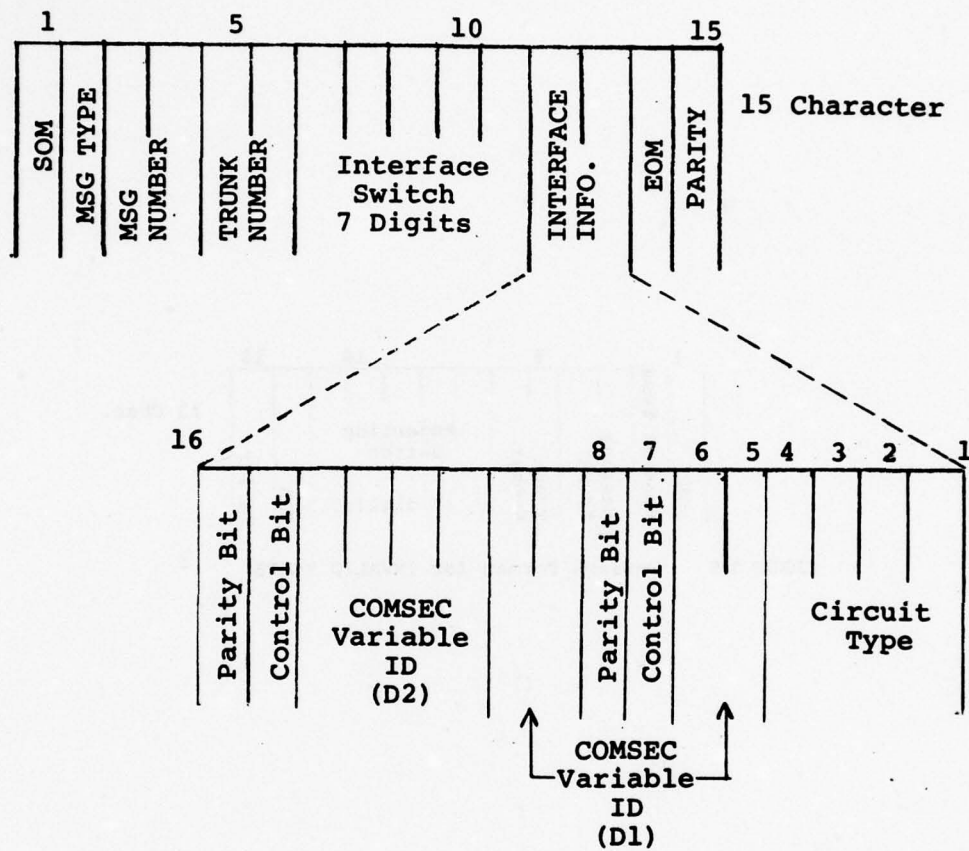


Figure 3.7 Message Format for INTERFACE COMPLETE

Processing non contiguous fields often requires additional operations. Since parity can be dropped, and since ADCCP permits transparent communications, all fields can be closed making them contiguous which also tends to keep fields within octet boundaries. (Address digits for example are neatly formatted as two BCD digits per octet.)

This modification represents a considerable departure from a "standard" format and would increase format conversion processing at gateways. For the above reasons, this modification was not included in the basic recommendation.

3.2.1.3 FUTURE MODIFICATIONS

It is noted that DCA is in the process of developing a standard CCIS format for Class I traffic which is a modified version of the TTC-39 formats. Any changes in size or contents of information fields should result in corresponding changes to this recommendation.

3.2.2 CONTROL MESSAGES FOR CLASS II TRAFFIC

At the present time no military packet systems are in operation. The ARPA system is an existing system and does provide basic information as to the required content of CCIS messages for Class II traffic. Other sources useful for determining the recommended formats are the Autodin II draft specification and the CCITT X.25 recommendation. However, both of these latter sources provide recommendations for the loop (DTE/DCE) interface but leave open the trunk or interoffice signalling format. Experience on the ADPT testbed node software development has also provided useful background for developing the recommended CCIS formats for Class II traffic. Tables 3.1 and 3.2 illustrate the formats recommended.

The following paragraphs describe the functions and formats of the messages and information fields of the CCIS data for Class II packets. It is noted that data may be appended to the basic CCIS message to reduce overhead and to speed data transmission.

TABLE 3.1 PACKET DATA CALL HEADER

1. Format Identifier
2. Packet Type/Security
3. Destination Node
4. Net Address (Destination Line)
5. Destination Line
6. Packet Sequence No.
7. Route Controls
8. Net/User Controls
9. Route Tabulation
10. Route Tabulation
11. Route Tabulation
12. Logical Channel
13. Precedence/Security
14. Transmission Control Code (TCC)
15. Orig. Node
16. Orig. Net/Orig. Line
17. Orig. Line
18. End of Header Symbol (EOH)

NOTE: This format applies to call request, call in progress, call terminate, and single packet call headers.

TABLE 3.2 PACKET RESPONSE

1. Format Identifier
2. Packet Type
3. Destination Node
4. Net Address/Destination Line
5. Destination Line
6. Allocation
7. Route Controls
8. Net/User Controls
9. Route Tabulation
10. Route Tabulation
11. Route Tabulation
12. Logical Channel
13. EOH

NOTE: This format applies to all data packet responses, including all specific acknowledgment and rejects which are uniquely identified by the "Packet Type" octet.

3.2.2.1 TRANSACTION DESCRIPTIONS

3.2.2.1.1 CALL REQUEST (SIMILAR TO CALL INITIATE)

This message is used to initiate a transaction and transmit the initial packet data. The message identifies the packet type, precedence and security data, calling party characteristics (if required) and originating node identification and logical channel. Data may be appended.

3.2.2.1.2 CALL IN PROGRESS PACKET

Similar to Call Request packet, however, packet identifier indicates that packet is a middle packet of a transaction. Data may appended.

3.2.2.1.3 CALL TERMINATE

This packet is same as a call progress packet, but indicates that this is the last packet of the transaction (call). Data may be appended.

3.2.2.1.4 SINGLE PACKET CALL

This packet is the same as a call request packet, but indicates that packet ends the transaction (call). Data may be appended.

3.2.2.1.5 PACKET ACKNOWLEDGEMENT

This is sent in response to a packet when the destination node has received one (or more) transmitted packets and has determined buffer availability. The response packet identifies the data call received, and indicates how many more packets can be delivered. A special flag is used to verify the special case where the acknowledgement also constitutes confirmation of a new call setup.

3.2.2.1.6 REQUEST FOR NEXT MESSAGE

This is sent in response to the last packet of a message and in lieu of normal packet acknowledgement when all packets of a message have been delivered to and acknowledged by the destination host. It may also used for connection breakdown and to inform the source host that the message was delivered to the destination host.

3.2.2.1.7 BUFFERS FULL

This is sent in response to a call request to indicate that the destination node cannot accept the data call due to lack of buffer space.

3.2.2.1.8 TABLES FULL

This is sent in response to a call request to indicate that destination node cannot accept the data call due to lack of connection table entry space.

3.2.2.1.9 CALLED LINE UNAVAILABLE

This is sent in response to a call request to indicate that all channels of the line requested are occupied.

3.2.2.1.10 CALLED LINE IS OUT OF SERVICE

This is sent in response to a call request to indicate that the called line is out of service.

3.2.2.1.11 INVALID ADDRESS

This is sent in response to any type call packet to indicate that the address line is not in use; therefore, the packet is rejected.

3.2.2.1.12 INVALID SECURITY

This is sent in response to any type call packet to indicate that the addressed line cannot accept the packet because the security designator of the packet exceeds the allowable security on that line. Packet is rejected.

3.2.2.1.13 INVALID SEQUENCE NUMBER

This is sent in response to any type call packet to indicate that the sequence number of the packet received is not allowed by the allocation and therefore the packet is rejected. First packets of a transaction must have sequence number zero.

3.2.2.2 TRANSACTION DESCRIPTORS

3.2.2.2.1 DATA PACKET HEADER FORMAT

This paragraph describes the specific functions and contents of each field which comprises each of the packet headers. Packet data call requests, call progress, call terminate, and single packet calls utilize the header format depicted in Table 3.1 and described below.

a) Format Identifier (octet 1)

A generic code. A unique code is used for all packets associated with Class II data traffic transmission. Code is a single octet in length. Class II Code = $\emptyset 2_{16}$

b) Packet Type (octet 2)

Indicates the specific type of Class II packet sent.

Field is 4 bits b1 - b4 coded as follows:

0_{16}	Call Request
1_{16}	Call Progress
2_{16}	Call Terminate
3_{16}	Single Packet
4_{16}	Packet Acknowledgement
5_{16}	Request for Next Message
6_{16}	Buffers Full
7_{16}	Tables Full
8_{16}	Called Line Unavailable
9_{16}	Called Line Out of Service
10_{16}	Invalid Address
11_{16}	Invalid Security
12_{16}	Invalid Sequence No.

c) Security (octet 2)

Bits 5-8 specifies security of packet. Coded as follows:

0_{16}	Special Intelligence
----------	----------------------

- 1₁₆ Top Secret
- 2₁₆ Secret
- 3₁₆ Confidential
- 4₁₆ Encrypted for Transmission only
- 5₁₆ Unclassified

d) Destination Code (octet 3)

Bits 1-8 provide capacity for up to 99 node designators employing BCD codes for each digit. It is recommended, however, that only codes 01-79 be used reserving designators 8 and 9 as user escape digits. Although a binary oriented format is slightly more efficient, use of BCD offers advantages which seem to outweigh the overhead penalty.

e) Network Address (octet 4)

Bit 1 designates a "dual homing" address. Bits 2-4 are used to designate eight destination network addresses and coded as follows:

- 0₈
- 1₈
- 2₈
- 3₈
- 4₈
- 5₈
- 6₈
- 7₈

Net Addresses (Including own Designator)

f) Destination Line (octet 4)

Bits 5-8 are BCD encoded and represent the hundredths digit in a destination line code which can range from 000-999.

Code.000 is reserved for data messages destined to the node.

Destination Line (octet 5)

Bits 1-4 and 5-8 represent the tens and units digit of the destination line coded in BCD.

g) Packet Sequence Number (octet 6)

Binary oriented code encoded to represent sequence numbers 0-255, i.e. 1st packet of a transaction is number 0, second packet is number 1, etc. Sequence number provides for accountability of packets during transaction delivery period.

h) Route Control Data (Forward Direction) Octet 7

Bit 1 and 2 indicate restrictions if any on satellite paths permitted.

- Ø No satellite
- 1 US Sat only
- 2 Spare
- 3 No restriction

Bits 3 and 4 indicate satellite hops which have actually occurred and are coded as follows:

- Ø No satellite
- 1 US Sat only
- 2 Spare
- 3 Other than US

Bit 5 is spare

Bits 6-8 are used to indicate any restriction or special instructions involved in routing the packets and are coded as follows:

- 0₈ No Restrictions
- 1₈ US only
- 2₈ US and this network only
- 3₈ This network only

4₈ Route to exit Switch

5₈ Route to Store and Forward Facility

i) Net/User Controls (octet 8)

Bits 1-4 are reserved for the source node (forward calls) and are used as follows.

Bit 1 - Request for Connection

This bit is set in a call request message if a new connection is requested for a packet other than the first packet in the message. It is used at the destination node to permit acceptance of a packet whose sequence number is other than zero. (This situation can occur as a result of flow control procedures.)

Bit 2 - Trace

If this bit is set all nodes receiving the packet report receipt of the packet to the net control center and local supervisor.

Bit 3 - Core

This bit indicates that data in the packet is a binary core image.

Bit 4 - Data Packet Flag

Indicates this packet is a data packet. Bits 5-8 are used to supply certain user information to the destination node.

Bits 5 and 6 are used to indicate a user speed category to provide information for flow control. Specific categories have not been defined; however, possible code is as follows:

0 ₄	0-9600	KBS
1	9600-19.2	KBS
2	19.2-38.4	KBS
3	Greater than 38.4	KBS

Bits 7 and 8 are reserved for future use such as indicating that terminal crypto of a specific type is to be employed.

j) Route Tabulation (octets 9, 10, 11)

A key problem in any network is avoidance of "ringaround". This group of 24 bits is utilized to prevent routing packets back to nodes that they have passed. Each digit of the destination address provides a level of routing as follows:

Network (1st address digit)
Node Group (2nd address digit)
Node Unit (3rd address digit)

Packet route rules are:

- a) Never route a packet back to a network it has left.
- b) Never route a packet back to a node group it has left.
- c) Never route a packet back to a node unit it has left.

The net bits indicate all the nets that have been traversed (if any). The nodes are grouped via the second address digit into up to eight area groups. The node group bits indicate what areas have been traversed within a given network. If a packet is traversing nets, this field is erased and reused each time a new net is entered.

The node unit digit bits indicate what nodes have been traversed within an area or group. If a packet is traversing areas (within a network) this field is erased and reused each time a new area is entered.

k) Logical Channel (octet 12)

Bits 1-8 provide 256 logical numbers for set up of

multiple virtual calls via each access line. Each line will, however, be limited as to maximum logical channel number useable depending on host characteristics.

- l) Precedence (octet 13)
Bits 1-4 provide 16 levels of precedence.
- m) Security (octet 13)
Bits 5-8 are redundant to security bits in octet 2.
- n) Transmission Control Code (octet 14)
Bits 1-8 provide 256 codes to provide a means to compartmentalize traffic and define controlled communities of interest among subscribers.
- o) Originating Node (octet 15)
Bit 1-8 represent two BCD digits indicating the node of origin.
- p) Originator Net Address (octet 16)
Bit 2-4 indicate the originating network.
- q) Originator line (octet 16)
Bits 5-8 are encoded and represent the hundredth digit in the originator line code. Range is 000-999. Code 000 is reserved for messages originated by the node.
- r) Origination Line (octet 17)
Bits 1-4 and 5-8 represent the tens and units digits of the originator line coded in BCD.
- s) End Header Symbol (EOH)
This symbol is a fixed format code used to indicate the end of the header. Code FF_{16} is suggested.

3.2.2.2.2 PACKET RESPONSE FORMAT

Table 2.2 presents the recommended format for the following CCIS packet response messages: Packet acknowledgement, request for next message, buffers full, tables full, called line unavailable, called line out of service, invalid address, invalid security, invalid sequence number. The format is discussed below:

- a) Format Identifier (octet 1)
Same code as packet header.
- b) Packet Type (octet 2)
Indicates the specific type of CCIS message. Field is 4 bits, b1-b4 coded as previously described for packet header format.
- c) Security (octet 2)
Bits 5-8 return the security field as presented in the data packet header.
- d) Destination Node (octet 3)
Two BCD digits indicate destination node of the response packet. It is noted that this octet is identical to the originating node code (octet 15) of that packet data header format which is being answered.
- e) Destination Net Address (octet 4)
Bits 2-4 indicate destination net address. It is noted that this field is identical to the originating net address field located in octet 16 of the packet data header format.
- f) Destination Line (octets 4 and 5)
Three BCD digits indicating destination line are identical to the originating line number codes in octets 16 and 17 of the data header format.
- g) Allocation (octet 6)
Indicates maximum data packet the source node is permitted to send and implies that buffer space has been allocated by the destination node. In the event of throttling, the allocation will be identical to the last packet sequence number transmitted by the source node.
- h) Route Control (octet 7)
Same as octet 7 of packet data format.
- i) Net/User Responses (octet 8)
Bit 1 - Connection Confirmed. Sent in response to a request for connection indicating connection is confirmed.

Bit 2 - Trace. If this bit is set all nodes receiving the packet report receipt of the packet to the net control center and/or the local node supervisor.

Bit 3 - Request new connection. Sent to indicate destination node is breaking down the existing connection. Subsequent packet must request new connection (throttling).

Bit 4 - Response Packet Flag. Indicates packet is response packet.

- j) Route Tabulation (octets 9, 10, 11)
Serve identical purpose as route tabulation field in data header format.
- k) Logical Channel (octet 12)
Same code as packet header.
- l) End Header Symbol (EOH) (octet 13)
Same code (FF_{16}) as for packet data header.

3.2.2.3 RATIONALE FOR RECOMMENDATIONS

This discussion will primarily concentrate on explaining the rationale for the recommended formats for Class II header traffic.

Suffice it to say in regard to Class I traffic that the need for standardization and use of proven procedures far exceed the need for CCIS format efficiency. CCIS overhead for a Class I call is much less than one percent of the voice channel bandwidth. It is also recognized that the recommended TTC-39 format is being modified and standardized by the DCA standards committee. Our basic recommendation is to accept standard CCIS formats for Class I traffic, but envelope the format with the ADCCP link protocol format to establish CCIS packets.

As regards Class II traffic, three factors key to the development of the recommended format are:

- a) Recognition of the fact that format compression is important since transmission efficiency is sensitive to header length for Class II traffic.

- b) Recognition that high packet thruputs will be required. It is important to develop a format which tends to reduce the nodal processing required.
- c) Recognition of the fact that it is very important to assure the packets are not delivered to a wrong destination.

In some instances satisfying one requirement meant relaxing another and judicious compromise was made. The format was compressed by utilizing for the most part binary oriented formats and avoiding extraneous or redundant information. The exceptions merit further discussion.

3.2.2.3.1 BCD ADDRESSING

BCD formats rather than binary are recommended for addressing because:

- a) Many of the addresses will be generated either by smart terminals or by network software serving simple terminals. Conversion to BCD from terminal source (such as ASCII) requires somewhat less processing.
- b) Total overhead penalty for this exception is approximately one octet for equivalent addressing capability or about 4% of the header overhead. This is not extremely significant.
- c) A digit breakdown permits a numbering plan to be established which simplifies routing control. (i.e. digits have significance such as network, node group, node unit).
- d) Redundancy inherent in the BCD code provides some protection against accidental generation of incorrect addresses.

3.2.2.3.2 FORMAT IDENTIFIER

The Format Identifier could, perhaps, be combined with the packet type identifier, however, since a unified node must handle multi-classes of traffic and may include special facilities such as store and forward message switching, it is important to rapidly assign incoming node packets to the appropriate major software module,

thus the format identifier serves as a first level function delineator. Four bits appear sufficient as length for this field, however, since the field is specified for the TTC-39 as a full octet, it is so specified for Class II data.

3.2.2.3.3 EXTRANEOUS OVERHEAD

The intent is to avoid placing information in each header not necessary for packet delivery; therefore, it is appropriate to discuss that information in the packet whose useage is not readily obvious or is open to legitimate debate.

3.2.2.3.3.1 NET ADDRESSES

Two half octets, one originating and one destination, were included in the format to permit addressing external networks that may exist at some future time. These could be eliminated by introducing different formats for packets leaving the network and using a different format identifier for these packets. The decision whether to take the 4% overhead penalty or add another format is subjective.

3.2.2.3.3.2 ROUTE CONTROL

Route control appears to be desirable to limit satellite usage, to control worst case delivery times, and possibly as a means to attain a degree of communications privacy. The decision whether inclusion of the data defined in the route control field is worth the overhead penalty (another 4%) is a valid issue. In a unified node ability to direct packets to the nearest S&F facility without altering the basic header format appears essential, therefore the "S&F" bit in the route control field is certainly needed in the header.

3.2.2.3.3.3 THE NODE CONTROL

The node control field is needed for transfer of flow control information and other net control data. Although specific bit recommendations are perhaps debatable, the need for at least one octet set aside for node and user inputs appears necessary.

3.2.2.3.3.4 ROUTE TABULATION

Route tabulation or other means of "ringaround" control is necessary. Since a key advantage of a packet communication is the ability of the packet to seek the destination via any of a multiplicity of possible routes, a method such as the bit mapping approach proposed (which is inherently permissive in routing, but prevents returning packets to nodes which have processed them) appears desirable.

3.2.2.3.4 REDUNDANT DATA

Redundancy where recommended serves either to aid packet processing or prevent false delivery.

3.2.2.3.5 FORMAT FEATURES WHICH AID PACKET PROCESSING

In order to aid packet processing the following features were incorporated in the recommended formats.

- 1) All data packets and packet responses involving Class II data locate those fields involving tandem packet processing in identical locations. The fields so mapped are:

Format Identifier

Net Control Field

Destination Address Fields

Route Controls

Route Tabulations

It is recognized the tandem processing of data packets will probably employ the precedence field for priority handling. It is, however, recommended that response packets always employ the highest priority. A bit in the net control field permits rapid differentiation of data and response packets.

- 2) Unique identifiers for format and packet type permit rapid branching to the desired software process routines.
- 3) Route controls aid the routing algorithm and permit special routing of those packets destined for other nets or processes.
- 4) Octet organization permits byte rather than more complex bit processing in the handling of many of the data fields.

3.2.2.3.6 FORMAT FEATURES WHICH AID PACKET PROTECTION

Packet protection via transmission links is afforded via the use of a 32 bit CRC check per the ADCCP protocol. It is extremely important to avoid delivery of a packet to the wrong destination especially where security is essential. Aside from the CRC check, it is necessary to assure the hardware and software malfunctions do not lead to misdelivery. It is not the intent in this writeup to recommend hardware and software protective features, but rather to assure that the recommended format has sufficient data to permit a rapid software check to be made at the delivery node, immediately prior to delivery, to assure that delivery is proper. Several features in the format that enhance such a checkout are:

- a) Format Identifier must be correct. Assuming only eight different identifiers are valid only one of thirty-two random data packets would pass this check.
- b) Security characters must match. Only one of sixteen random data packets would pass this check.
- c) Address must be valid. BCD characters have sufficient redundancy, such that only 1 out of ten random data packets could pass this check.
- d) Sequence number must be valid. Only one of 256 random data packets could pass this check.

The above checks may prove to be a sufficient safeguard, however, use of an "End of Header" character provides for a further packet screen which can be employed in addition to or as a substitute for other checks. It is not felt that software parity checks are needed considering all the safeguards existing, however, hardware parity checks on memory readouts appear to be a desirable processor feature to protect against transient bit errors which could occur during memory transfers.

3.2.2.3.7 REASON FOR "COMPRESSED RESPONSE PACKET

A packet response is required for every packet sent during periods of congestion where multi-buffer allocations must be avoided. Thus,

the packet response represents a considerable portion of packet data overhead. Employing a response packet which is shorter in length than the data packet header improves efficiency. The response packet does not need to indicate originating node or line since the source node receives the response and can assume that the response came from the designated data destination. Precedence/category and TTC codes can be dropped as well. (Response packets are treated as high priority messages.) Logical channel number must be returned to permit the source node to identify the data call. Since each data call is uniquely identified by originating node, originating line, and logical number, the destination line of the response packet must always indicate the originating line of the data sent in the forward direction.

3.2.2.4 POSSIBLE USE OF SINGULAR TRANSACTION IDENTIFIERS (TID)

It may prove desirable to exchange unique node transaction identifiers (TID). The TID field would be a field large enough to accommodate the maximum number of simultaneous connection permitted by any node (e.g. 12 bits = 4096 binary). During the initial connection setup the first packet exchange would exchange TIDs to set up the source destination connection (e.g. SOURCE TID 25 DESTINATION TID 457). Subsequent data packets would use the foreign TID in place of line addressing and need not contain any return address identification. This approach simplifies locating the bookkeeping block established for each connection, and may considerably simplify table management.

3.2.2.5 ADDENDUM

The recent release of the Autodin II Contract has introduced another factor not previously considered. The Autodin II

specification dated November 1975 specifies a binary segment leader (See Fig 3.8). Future hosts and intelligent terminals interfacing Autodin II will employ this segment leader and the unified node should also be compatible with these hosts and terminals. In effect a defacto standard has been established. The leader is a subset of the packet header previously described in this section, therefore all corresponding fields of the proposed header must be modified and relocated so they are located in the same relative locations as shown in Figure 3.8 and must correspond in format (i.e., binary) and in format codes and meanings (refer to Autodin II specification).

It is further noted that the logical address field will not be employed for the multiple channel interface (source - destination address combinations are used in place of unique logical numbers). Therefore, this field can be dropped from the packet header. Recommendations for fields other than the leader field which may be part of the unified node packet headers (e.g. net/user controls, route tabulations, allocations, route controls, format identifiers, packet type) are unaffected except that it now appears reasonable to employ binary address codes throughout.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
+0	P R I	CATEGORY				SPARE						SECURITY				
+1	TRANSMISSION CONTROL CODE					SPARE		COMMAND CONTROL CODE								
+2	SECURITY		TRANSMISSION CONTROL CODE								SPARE					
+3	DESTINATION IDENTIFICATION LOGICAL ADDRESS															
+4	TCP VERSION NUMBER		SEGMENT NUMBER													

FIGURE 3.8

BINARY SEGMENT LEADER

4.0 INTERFACING THE ADPT TESTBED TO EXISTING NETWORKS

A requirement of this study effort is to determine how the automatic digital switch/telecommunication testbed can interface to Autodin I and ARPA networks. It is important to note that this study is not aimed at developing interfaces for the future unified node. Under this contract a node simulator has been developed which performs many of the functions of a unified node as a small scale brassboard. This simulator is described in detail in Section 5. The thrust of this study is to determine how the simulator could be modified so that user terminals of the simulator can interface with ARPA and Autodin I networks. The proposed ARPA net interface approach was confined to a study effort. The Autodin I interface was implemented, but not certified. The Autodin interface existed in the testbed prior to this phase of the program. Specifically a psuedo host or gateway was implemented in this phase, so that the packet switch data could be transferred to the existing message switch hardware and vice versa.

The testbed simulator interfaces Autodin as either a Mode I or Mode II subscriber.

Since it is uncertain whether the testbed would be in physical proximity with an ARPANET Interface Message Processor (IMP) or Terminal Interface Processor (TIP) the recommended interface to the ARPANET places the testbed simulator in the more general role of a distant host to an IMP or TIP. However, the simplest interface to implement would place the simulator in the role of a terminal subscriber to a TIP. Therefore, if in fact the testbed is colocated or near a TIP this approach should be considered.

The TIP terminal interface is easy to implement since the testbed hardware can under program control support an asynchronous ASCII interface. The TIP terminal interface would function for data transfer similar to the node II autodin interface, however the ARPANET specific user procedures would be employed. A software modification of the testbed would be needed to support the transfer of data.

The following ARPANET interface discussion applies primarily to the more general case of interfacing to an IMP or TIP as a distant host, but does describe user protocol that would be equally applicable should the TIP terminal interface approach be employed.

4.1 ADPT OPERATION AS A HOST OF THE ARPA NETWORK

It has been assumed that the ADPT would act as a very distant HOST (as defined herein) and would be assigned to operate with a conveniently located IMP, Interface Message Processor, in accordance with rules already established in the ARPA network for use with a very distant HOST, as described herein.

Figure 4.1 shows the ADPT Host Interfaces to the ARPA network that would be required; these include the software programs, also, with references to appropriate ARPA Network reports and some selected papers for program details. References cited throughout this section are listed at the end of the section. The ADPT Host would be controlled locally by TTY and/or VDT terminals. These terminals would operate via local ADPT TTY/VDT programs. Operation with remote ARPA Net serving HOST TTY/VDT programs would require some new ADPT software to be interoperable with a variety of TTY and VDT terminals and conventions employed in the ARPA network.

The user Telecommunication Network (TELNET) protocol/program provides for connections to a serving Host and for message exchange with the serving Host.

The HOST-HOST Protocol shown next in the figure provides for process control, socket selection, formation of socket connections, process interchanges and user identification and log-in interactions

The Network Control Program (NCP) establishes actual connections with the serving Host via the IMP associated with the ADPT Host. It controls the flow of information, establishes the Leader (host-to-IMP), interchanges of Host-to-IMP and IMP-to-Host information

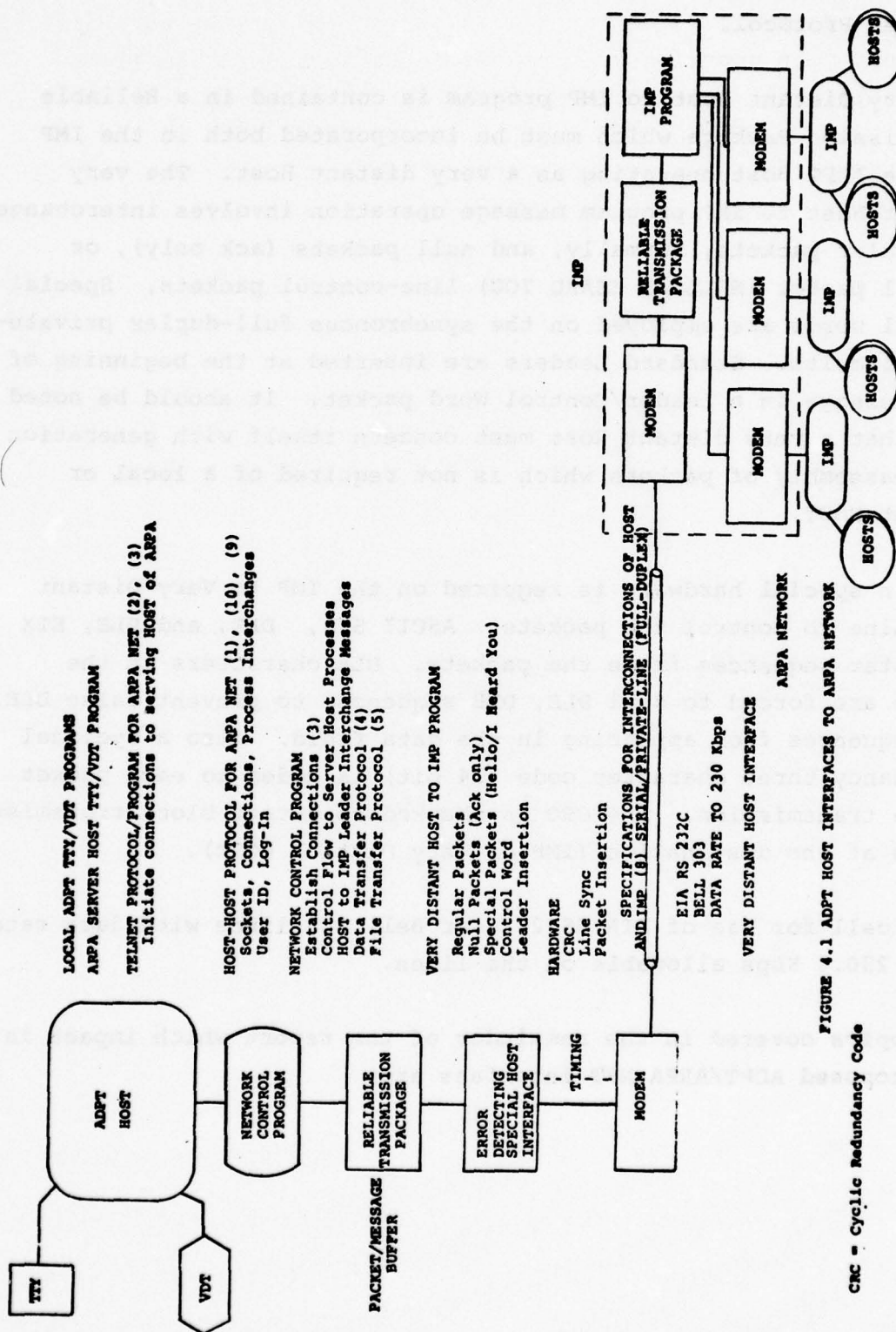


FIGURE 4.1 ADPT HOST INTERFACES TO ARPANET

CRC = Cyclic Redundancy Code

and control messages, and controls Data Transfer Protocol and File Transfer Protocol.

The Very Distant Host to IMP program is contained in a Reliable Transmission Package which must be incorporated both in the IMP and the ADPT Host operating as a very distant Host. The very distant Host to IMP program message operation involves interchange of regular packets, normally, and null packets (ack only), or special packet (HELLO/I HEARD YOU) line-control packets. Special control words are employed on the synchronous full-duplex private-line circuits. Standard Leaders are inserted at the beginning of each message in a Leader/Control Word packet. It should be noted here that a very distant Host must concern itself with generation and reassembly of packets which is not required of a local or distant HOST.

Certain special hardware is required on the IMP to Very Distant Host Line to control the packets. ASCII SYN, DLE, and DLE, ETX character sequences frame the packets. DLE characters in the packet are forced to dual DLE, DLE sequences to prevent false DLE, ETX sequences from appearing in the data field. Also a cyclical redundancy three character code (24 bit) is added to each packet before transmission. The CRC is checked to detect block transmission errors at the destination (IMP or Very Distant HOST).

Specs call for use of EIA RS 232C or Bell 303 lines with data rates up to 230.4 Kbps allowable on the lines.

The topics covered in the remainder of the report which impact in the proposed ADPT/ARPA NET interface are:

- . Host-Host Protocol
- . Telecommunication Network (TELNET) Protocol
- . Network Control Program (NCP)
 - The Logging-In Process
 - User Identification
- . ARPA Net Leader Format
- . IMP Header Format
- . Local/Distant/Very Distant Host Interfaces
- . Special Provisions for Very Distant Host
 - Reliable Transmission Package
 - Error Detecting Special Host Interface Packet Format On Line
- . Appendix A Host-to-IMP and IMP-to-Host Leader Formats
- . Appendix B ADPT Simulator Leader/Header Format

4.1.1 HOST-HOST PROTOCOL

Host-to-Host protocol in the ARPA Network is the highest level control arrangement and procedure employed to communicate between user jobs or processes within the Host programs. A process requiring communication with another process at a remote Host makes requests of its local controlling or executive program to establish the desired connection using the Host-to-Host protocol.

Basic commands, termed "primitives," provide the mechanism for establishing the one-way byte-oriented connections. A connection consists of a simplex link between a send socket and a receive socket, between the two Host processes. See Figure 4.2. Normally, two separate connections are set up to achieve two-way communication between the processes.

Primitives available at the user-level Host interface include:

- . Initiate Connection (Local socket, foreign socket)
- . Wait for Connection (Local socket)
- . Send, Receive (local socket, data)
- . Close (local socket)
- . Send Interrupt signal (local socket)

4.1.1.1 HOST-TO-HOST DESIGN PHILOSOPHY

It was assumed in establishing the design philosophy that the individual Host computers in the network would have entirely independent purposes. Thus, it has been desirable to preserve entirely decentralized administrative control of these Host computers. Each Host time-sharing program supervisor has its own accounting and resource allocation mechanisms - and controls its work load from the network in the same manner as it controls its local processing load.

Basic interactive mechanisms are facilitated in the system operation.

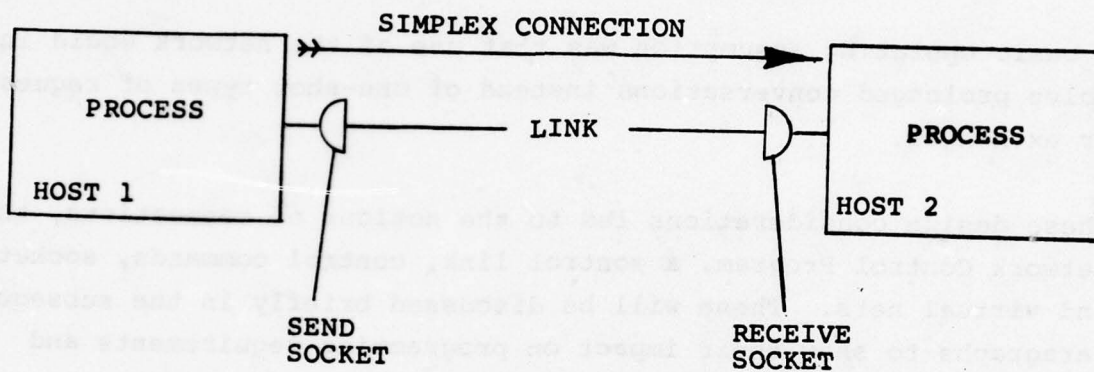


FIGURE 4.2

HOST-TO-HOST

CONNECTION BETWEEN PROCESSES

Restrictions of character sets, programming languages, etc. were avoided in the system design.

The design of the network operation was kept open-ended to allow for changes, and particularly improvements in connections as the need arises.

The impact of the network software in Host software was kept to a minimum.

A basic operating assumption was that use of the network would involve prolonged conversations instead of one-shot types of requests or exchanges.

These design considerations led to the notions of connections, the Network Control Program, a control link, control commands, sockets, and virtual nets. These will be discussed briefly in the subsequent paragraphs to show their impact on programming requirements and system procedures.

4.1.2 TELECOMMUNICATION NETWORK (TELNET) PROTOCOL

Processes utilizing Host-to-Host protocol can create connections to other processes and transmit messages in the form of bytes.

For remote calls, a Telecommunications Network (TELNET) protocol supplements the processes control. A user at a terminal serviced by his Host establishes control of a process in a remote Host just as though he were a local user of the remote Host.

This is done by performing the following functions through his TELNET user processes (User-TELNET).

TELNET Commands (Representative)

Indicate pair of connections to serving Host (serving host is the remote host).

- . Send messages to serving Host
- . Receive messages from serving Host
- . Send Host-to-Host interplay signal as appropriate
- . Terminate connections

The user-TELNET distinguishes between commands to be acted upon locally and commands intended for the Serving Host. An escape character is reserved to mark the local commands.

The user-TELNET functions are normally provided by a user-level program in the using Host. Related functions, such as filing of records, may be provided in addition.

Figure 4.3 shows the representative interplay (10) for a user at UTAH who wishes to operate the Conversational Algebraic Language subsystem on the XD5-940 at SRI.

A user at the University of Utah (UTAH) is sitting at a teletype dialed into the University's PDP-10/30 time-sharing system. He wishes to operate the Conversational Algebraic Language (CAL subsystem on the XDS-940 at Stanford Research Institute (SRI) in Menlo Park, California. A typical TELNET dialog is illustrated in Figure 4.3. The meaning of each line of dialog is discussed here.

- 1) The user signs in at UTAH.
- 2) The PDP-10 run command starts up the TELNET subsystem at the user's HOST.
- 3) The user identifies a break character which causes any message following the break to be interpreted locally rather than being sent on to the foreign HOST.
- 4) The TELNET subsystem will make the appropriate system calls to establish a pair of connections to the SRI logger. The connections will be established only if SRI accepts another foreign user.

- (i) . LOGIN cr
- (ii) . R TELNET cr
- (iii) ESCAPE CHARACTER IS *cr
- (iv) CONNECT TO SRI cr
- (v) @ENTER CARR. cr
- (vi) @CAL. cr
- (vii) CAL AT YOUR SERVICE
- (viii) READ FILE FROM NETWRK. cr
- (ix) *NETWRK: DSK:MYFILE.CAL cr

A typical TELNET dialog
Underlined characters are those
typed by the user

FIGURE 4.3

The UTAH user is now in the pre-logged in state at SRI. This is analogous to the standard teletype user's state after dialing into a computer and making a connection but before typing anything.

- 5) The user signs into SRI with a standard login command.

Characters typed on the user's teletype are transmitted unaltered through the PDP-10 (user HOST) and onto the 940 (serving HOST). The PDP-10 TELNET subsystem will have automatically switched to full-duplex, character-by-character transmission, since this is required by SRI's 940. Full duplex operation is allowed for by the PDP-10, though not used by most Digital Equipment Corporation subsystems.

- 6) and 7) The 940 subsystem, CAL, is started. At this point, the user wishes to load a CAL file into the 940 CAL subsystem from the file system on his PDP-10.
- 8) CAL is instructed to establish a connection to UTAH in order to receive the file. "NETWRK" is a predefined 940 name similar in nature to "PAPER TAPE" or "TELETYPE".
- 9) Finally, the user types the break character (#) followed by a command to his PDP-10 TELNET program, which sends the desired file to SRI from Utah on the connection just established for this purpose. The user's next statement is in CAL again.

4.1.3 NETWORK CONTROL PROGRAM (NCP)

Processes within Hosts communicate with remote Hosts and with Network elements via a Network Control Program (NCP). In most Hosts the NCP is a part of the executive program, so the processes use system calls to communicate with it. The primary function of the NCP is to establish connections, break connections, switch connections, and control flow. (Figure 4.4).

To do this, NCP's must communicate. This is done via a control link over which control commands are passed. As an example, one command is used to assign a link and initiate a connection; a second, provides notification that a connection has been terminated.

A data transfer protocol is then employed to transfer data.

4.1.3.1 NCP SYSTEM CALLS

Here we sketch the mechanics of establishing, switching and breaking a connection. As noted above, the NCP interacts with user processes via system calls and with other NCPs via control commands. We therefore begin with a partial description of system calls and control commands.

System calls will vary from one operating system to another, so the following description is only suggestive. We assume here that a process has several input-output paths which we will call ports. Each port may be connected to a sequential I/O device, and while connected, transmits information in only one direction. We further assume that the process is blocked (dismissed, slept) while transmission proceeds. The following is the list of system calls:

Init <port> <AEN 1> <AEN 2>
 <foreign socket>

where <port> is part of the process issuing the Init

 <AEN 1>
and are 8-bit AEN's (See Figure 3)
 <AEN 2>

<foreign socket> is the 40-bit socket name of the distant end of the connection.

The first AEN is used to initiate the connection; the second is used while the connection exists.

The low-order bits of <AEN 1> and <AEN 2> must agree, and these must be the complement of the low order bit of <foreign socket>.

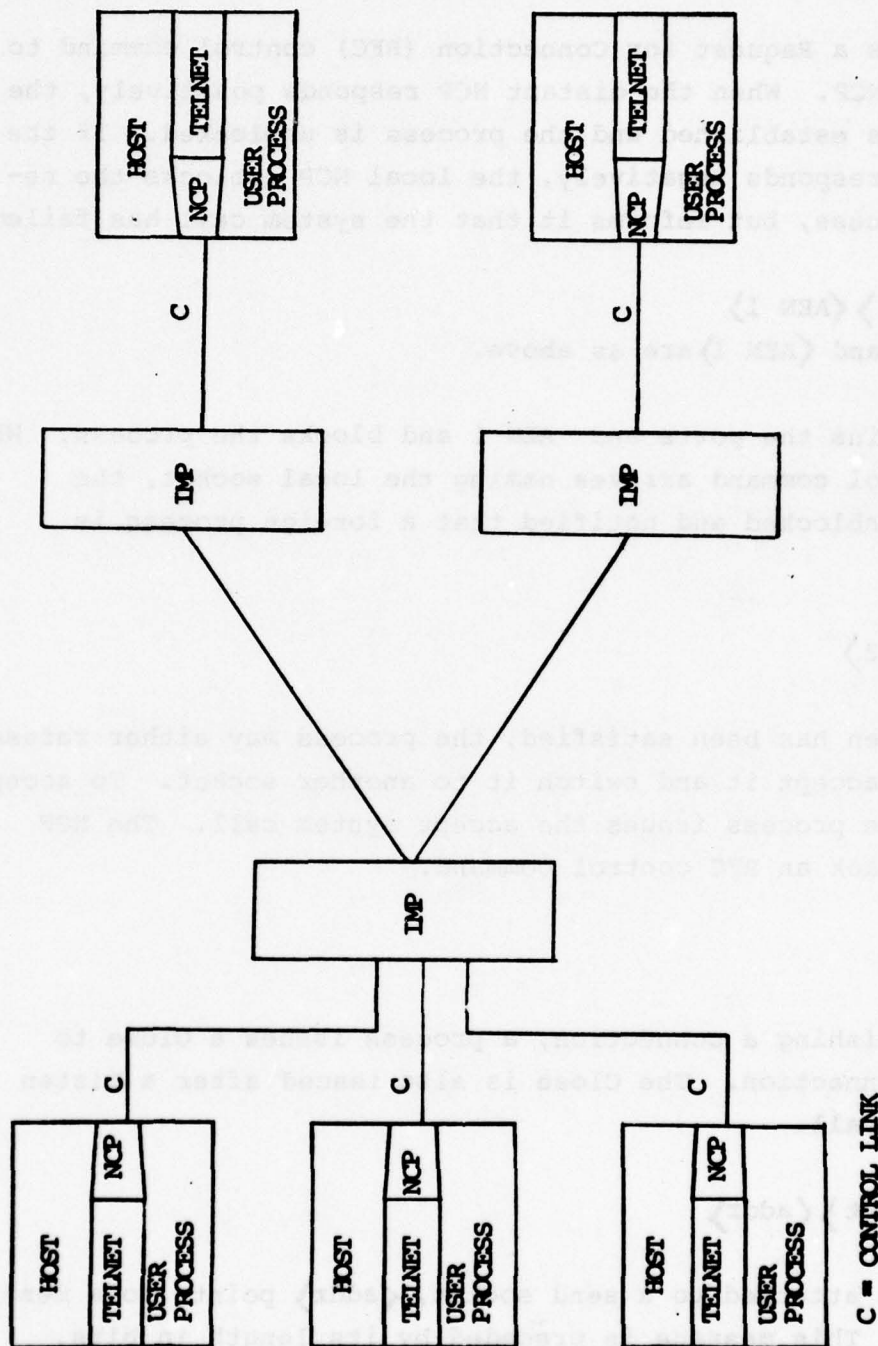


FIGURE 4.4 - NCP AND TELNET NETWORK CONTROL

The NCP concatenates <AEN 1> and <AEN 2> each with the user number of the process and the HOST number to form 40-bit sockets.

It then sends a Request for Connection (RFC) control command to the distant NCP. When the distant NCP responds positively, the connection is established and the process is unblocked. If the distant NCP responds negatively, the local NCP unblocks the requesting process, but informs it that the system call has failed.

Listen <port> <AEN 1>
where <port> and <AEN 1> are as above.

The NCP retains the ports and AEN 1 and blocks the process. When an RFC control command arrives naming the local socket, the process is unblocked and notified that a foreign process is calling.

Accept <AEN 2>

After a listen has been satisfied, the process may either refuse the call or accept it and switch it to another socket. To accept the call, the process issues the accept system call. The NCP then sends back an RFC control command.

Close <port>

After establishing a connection, a process issues a Close to break the connection. The Close is also issued after a Listen to refuse a call.

Transmit <port> <addr>

If a port is attached to a send socket, <addr> points to a message to be sent. This message is preceded by its length in bits.

If `<port>` is attached to a receive socket, a message is stored at `addr`. The length of the message is stored first.

4.1.3.2 CONTROL COMMANDS

A vocabulary of control commands has been defined for communication between Network Control Programs. Each control command consists of an 8-bit operation code to indicate its function, followed by some parameters. The number and format of parameters is fixed, for each operation code. A sequence of control commands destined for a particular HOST can be packed into a single control message.

```
RFC  <my socket 1> <my socket 2>  
      <your socket> <link>
```

This command is sent because a process has executed either an `Initiate` system call or an `Accept` system call. A link is assigned by the prospective receiver, so it is omitted if `<my socket 1>` is a send socket.

There is distinct advantage in using the same commands both to initiate a connection (`Init`) and to accept a call (`Accept`). If the responding command were different from the initiating command, then two processes could call each other and become blocked waiting for each other to respond. With this scheme no deadlock occurs and it provides a more compact way to connect a set of processes.

```
CLS   <my socket> <your socket>
```

The specified connection is terminated

```
CEASE <link>
```

When the receiving process does not consume its input as fast as it arrives, the buffer space in the receiving HOST is used to queue the waiting messages. Since only limited space is generally available, the receiving HOST may need to inhibit the sending HOST

from sending any more messages over the offending connection. When the sending HOST receives this command, it may block the process generating the messages.

RESUME <link>

This command is also sent from the receiving HOST to the sending HOST and negates a previous CEASE.

4.1.4 THE LOGGING-IN PROCESS

A process is provided in each Host which can accommodate the logging-in of users. This process is termed the "logger", and is organized to be a part of a vertical net which is assigned a user number of zero.

The "logger" is programmed to monitor calls on socket number zero. On receiving a call, on socket zero, the logger assigns it an even numbered socket and refuses a call to the socket numbered one less than the send socket originally calling. In this fashion the logger can initiate 127 conversations.

An illustration (10) of this procedure for a user identified as X'010005' (user number 5 at UCLA) signing in at UCLA with a start up of one of his programs there, will be described. User number 5 starts his program which wants to start a process at SRI. No process at SRI except the logger will respond to the UCLA user, so he executes "Initiate". The procedure continues until the SRI logger has completed the transaction with the UCLA process, the logger interrogates the UCLA process and if conditions are satisfactory the SRI logger will create a new process at SRI. This new process is tagged with the user number X'010005' and both connections will be switched to the new process.

4.1.4.1 LOGGING-IN EXCHANGE

Init, <port = 1><AEN 1 = 7>
 <AEN 2 = 7>
 <foreign socket = 0>

His process is blocked, and the NCP at UCLA sends

```
RFC (my socket 1) = X'0100050107'  
      (my socket 2) = X'0100050107'  
      (your socket) = X'0000000200'
```

The logger at SRI is notified when this message is received, because it has previously executed

```
Listen <port> = 9, (AEN 1) = 0
```

The logger then executes

```
Accept (AEN 2) = SS
```

In response to the Accept, the SRI NCP sends

```
RFC (my socket 1) = X'0000000200'  
      (my socket 2) = X'0000000258'  
      (your socket) = X'0100050107'  
      (link) = 37
```

where the link has been chosen from the set of available links.

The SRI logger then executes

```
Init (port) = 10  
      (AEN 1) = 59, (AEN 2) = 89  
      (foreign socket) = X'0100050106'
```

which causes the NCP to send

```
RFC (my socket 1) = X'0000000259'  
      (my socket 2) = X'0000000259'  
      (your socket) = X'0100050106'
```

The process at UCLA is unblocked and notified of the successful Init. Because the SRI logger always initiates a connection to the AEN one less than it has just been connected to the UCLA process then executes.

```
Listen (port) = 11  
      (AEN 1) = 6
```

and when unblocked,

```
Accept (AEN 2) = 6
```

(24 bits)	(8 bits)	(8 bits)
USED NUMBER	HOST NUMBER	AEN

FIGURE 4.5

SOCKET NUMBER ELEMENTS
(RECEIVE OR SEND SOCKET)

AEN = Another Eight
Bit Number

4.1.5 USER IDENTIFICATION

Each user in the ARPA Network is assigned a 32 bit number which uniquely identifies him in the network. This user number generally will consist of the 8-bit Host number of his home Host, preceded by a 24-bit user number, identifying him with the home Host. (Figure 4.5)

An alternate procedure can be utilized wherein the user signs-on at a Host, and his processes then are tagged with his user number. If he signs on a foreign Host, via the network, his number is used to tag processes he creates in the foreign Host. Thus, he can identify processes in a number of Hosts and can interconnect them in virtual nets.

There are 128 send sockets and 128 receive sockets for each user number at each Host, which can be employed by the user.

The Host-to-Host protocol makes provision for 128 sockets for each Host user. However, the protocol does not specify their useage with respect to process interconnection.

The convention for establishing communication between a user TELNET and a server-TELNET is provided by an Initial Connection Protocol (ICP) (3) for example:

- . Connection initiated by user-TELNET receive socket to serving Host's socket (a send socket)
- . When initial connection is established, the serving Host sends a generated socket number and closes the connection. This socket number identifies an adjacent socket pair at the serving Host through which the user-TELNET can communicate with the server-TELNET.
- . TELNET connections are then initiated between the now specified pairs of sockets. Two connections are used to provide bi-directional communication.

Note that the socket 1 at the serving Host is in use only long enough to send another socket number with which to make actual service connections indicated.

4.1.6 ARPA NET LEADER FORMAT

"Regular" messages are used for communication between IMPs and local or distant Hosts. These regular messages vary in length between 32 bits and 8095 bits.

Other message types provided for are listed in Table 4.1 as generated (a) by the Host, and (b) by the IMP.

The first 32 bits of a message are a leader which provide the control information and specify the type of message involved. Figure 4.6a shows the detail format of a Host-to-IMP leader, while Figure 4.6b shows the format for an IMP-to-Host Leader. Note that a 12 bit message ID is employed and that the Host sends the IMP the destination (8-bits) while the IMP sends the source IMP and particular Host involved at the source IMP location.

4.1.7 IMP HEADER FORMAT

The IMP leader format is depicted in Figure 4.7. This format is used for transfer of packets between IMPs of the ARPA network.

The IMP reformats information received from HOSTS in the ARPA Net Leader Format to obtain the format of Figure 4.7.

The maximum packet length employed is approximately 1000 bits.

4.1.8 LOCAL/DISTANT/VERY DISTANT HOST INTERFACES

LOCAL HOST

A local Host is connected to the associated IMP by a 30 foot multi-pair cable. The length of this cable is limited by the characteristics of the cable driver in the IMP.

Host-to-IMP Leader Format

4.6a

1	2	3	4	5	8	9	16	17	28	29	32
PRIORITY	FOR IMP	TRACE	OCTAL	MESSAGE TYPE	DESTINATION			MESSAGE-ID			SUB-TYPE

IMP-to-Host Leader Format

4.6b

1	2	3	4	5	8	9	16	17	28	29	32	
FROM IMP	FROM IMP		OCTAL	MESSAGE TYPE	SOURCE			MESSAGE-ID				SUB-TYPE

FIGURE 4.6 ARPA NET LEADER FORMAT

TABLE 4.1

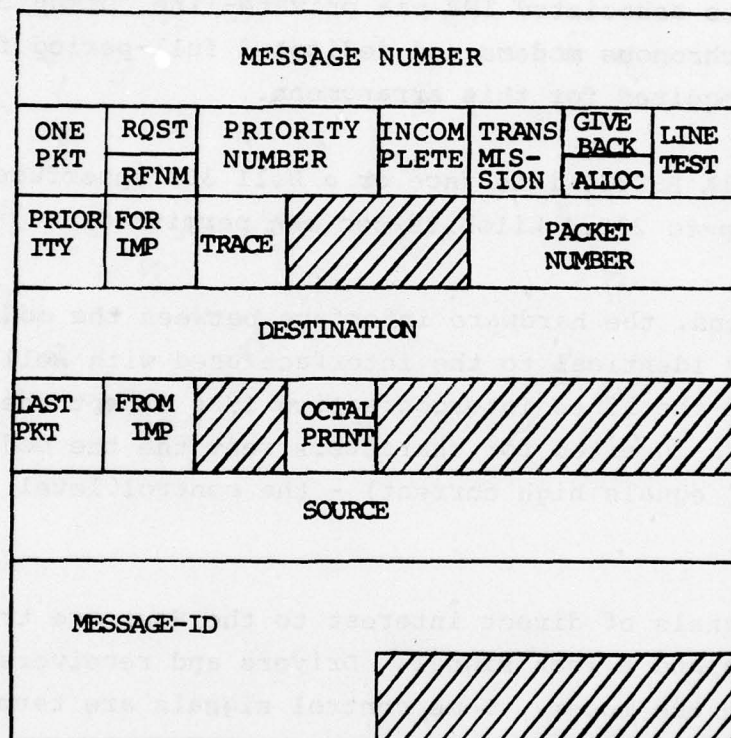
Message Types in ARPA Network:

Host to IMP

- Regular Messages
- Error Without Message Identification
- Host Going Down
- Uncontrolled Packet
- NOP
- Error with Message Identification

IMP to Host

- Regular Message
- Error in Leader (3 sub-types)
- IMP Going Down
- Uncontrolled Packet
- NOP
- RFNM
- Dead Host Status (14 types)
- Destination Host or IMP Dead (4 sub-types)
- Error in Data
- Incomplete Transmission (6 sub-types)
- Interface Reset



← 8 BITS →

IMP HEADER FORMAT

FIGURE 4.7

DISTANT HOST

A distant Host may be located up to 2000 feet from its associated IMP. For this arrangement, the Host personnel must provide a special interface that is compatible with the IMP, and must supply the multi-pair interconnecting cable (8).

VERY DISTANT HOST

A very distant Host is located at a distance of greater than 2000 feet from its associated IMP via private-line communication circuits. Synchronous modems and dedicated full-period full-duplex lines are required for this arrangement.

Either an EIA RS232C interface or a Bell 303 interface can be used. Speeds of up to 230.4 kilobits/sec are permitted.

At the IMP end, the hardware interface between the modem and the IMP is logically identical to the interface used with Bell 303/50 Kilo-bit modem on the lines interconnecting IMPs except the mark space convention is inverted for characters sent the the modem (i.e., binary "one" equals high current) - the control level lines are not inverted.

The only signals of direct interest to the Host are transmit and receive data and a lock signal. Drivers and receivers must be compatible with the modem. Some control signals are terminated in the modem or elsewhere, as appropriate.

4.1.9 SPECIAL PROVISIONS FOR DISTANT HOST

To implement the special provisions of the very distant Host interface to the full-duplex private-line, two special software packages are employed. See Figure 4.8

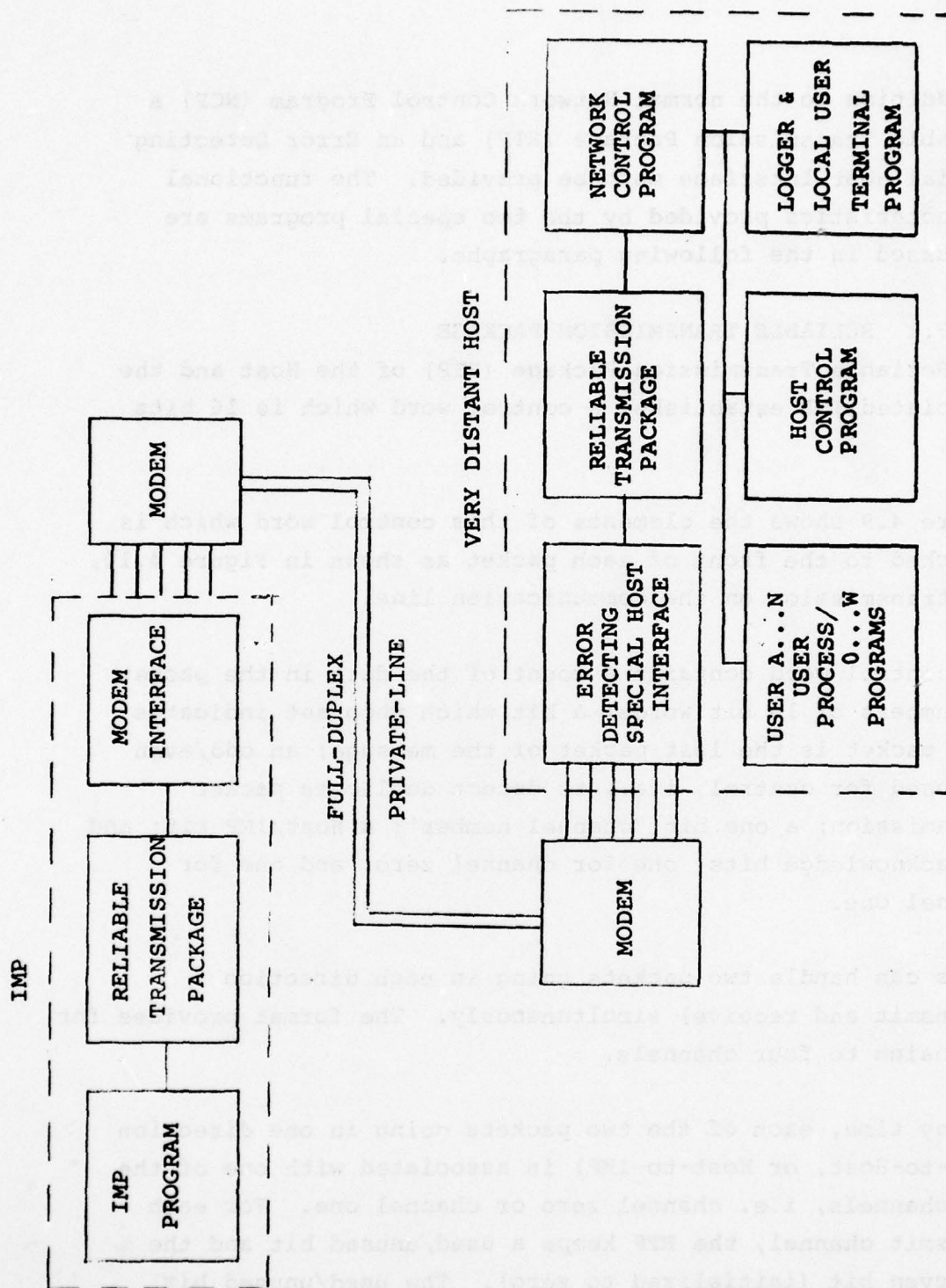


FIGURE 4.8 - IMPLEMENTATION OF VERY DISTANT HOST TO IMP INTERFACE

In addition to the normal Network Control Program (NCP) a Reliable Transmission Package (RTP) and an Error Detecting Special Host Interface must be provided. The functional characteristics provided by the two special programs are discussed in the following paragraphs.

4.1.9.1 RELIABLE TRANSMISSION PACKAGE

The Reliable Transmission Package (RTP) of the Host and the associated IMP establishes a control word which is 16 bits long.

Figure 4.9 shows the elements of this control word which is attached to the front of each packet as shown in Figure 4.10, for transmission on the communication line.

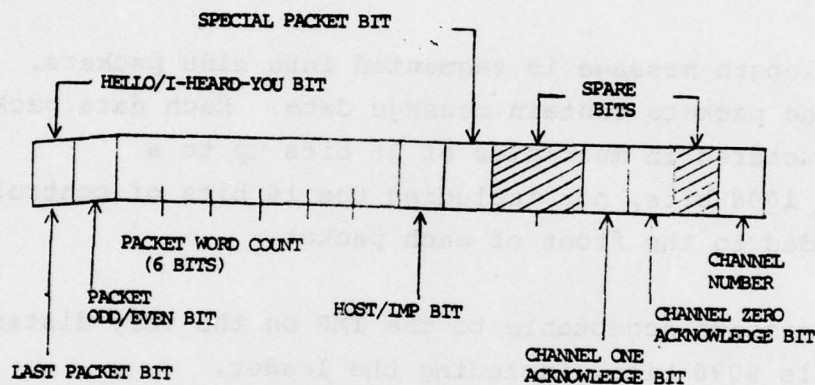
The control word contains a count of the data in the packet in numbers of 16 bit words; a bit which when set indicates this packet is the last packet of the message; an odd/even bit used for control, i.e., to detect duplicate packet transmission; a one bit "channel number": a Host/IMP bit; and two acknowledge bits, one for channel zero, and one for channel one.

RTP's can handle two packets going in each direction (transmit and receive) simultaneously. The format provides for extension to four channels.

At any time, each of the two packets going in one direction (IMP-to-Host, or Host-to-IMP) is associated with one of the two channels, i.e. channel zero or channel one. For each transmit channel, the RTP keeps a used/unused bit and the odd/even bit (initialized to zero). The used/unused bit indicates whether there is currently a packet associated with the channel. Note that packets must be retransmitted until

acknowledged. To allow for transmission delay, a "trial" delay before retransmission of 100 msec is suggested. This retransmission delay characteristic is to be adjustable.

Null packets are 16 bit packets containing acknowledgements when no further traffic is available. Null packets have a word count of zero.



REGULAR PACKET

NULL PACKET

SPECIAL PACKET

CONTROL WORD PLUS PACKET: 1ST PACKET HAS LEADER

16 BIT PACKET WITH ACK

16 BIT PACKET WITH SPECIAL PACKET BIT SET TO ZERO

HELLO/I HEARD YOU

HOST/IMP

FIGURE 4.9 - CONTROL WORD FORMAT FOR USE ON VERY DISTANT
HOST COMMUNICATION LINE

It is interesting, therefore, that quite unlike a local Host or a distant Host, a very distant Host must be aware of packets. Furthermore, the normal leader used also by local and distant Hosts is transmitted to a very distant Host (and to the associated IMP) in a separate packet which precedes the rest of the packets of the message. See Figure 4.10. Note that the control word is sent with this leader in the first packet, and in each additional packet of the message.

A maximum length message is segmented into nine packets. Eight of the packets contain message data. Each data packet can be structured in multiples of 16 bits up to a maximum of 1008 bits, not including the 16 bits of control word appended to the front of each packet.

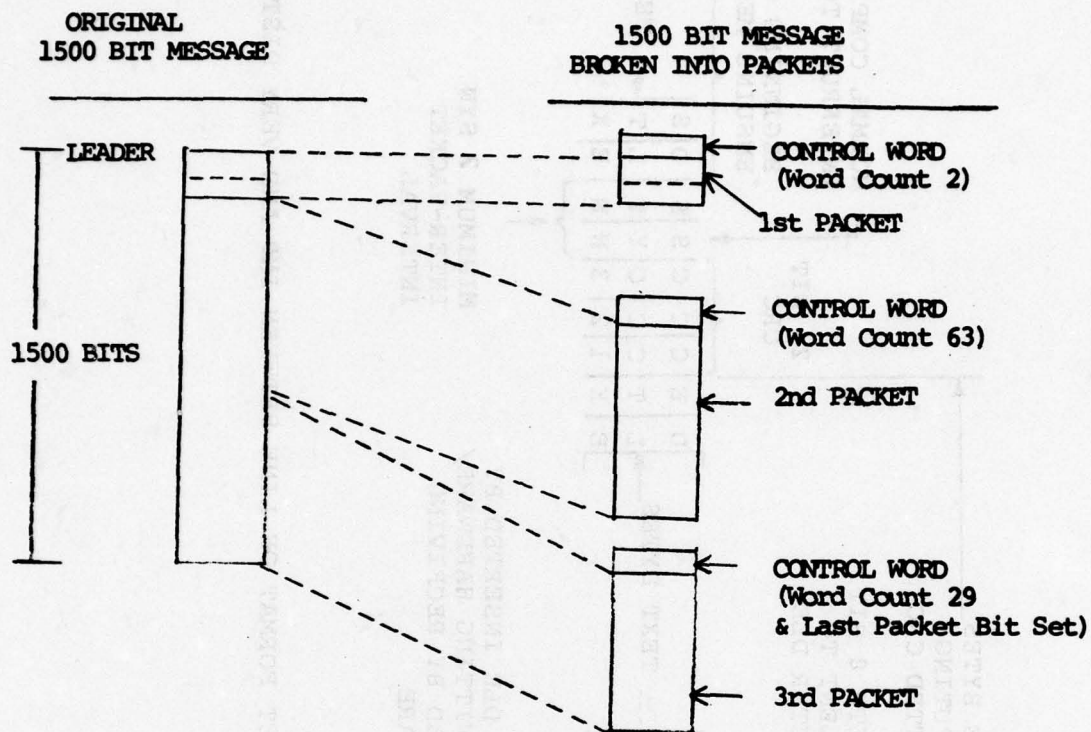
The total message acceptable to the IMP on the very distant Host line is 8096 bits, including the leader.

4.1.9.2 ERROR DETECTING SPECIAL HOST INTERFACE PACKET FORMAT ON LINE

Figure 4.11 depicts a packet sequence on the line for a very distant Host-to-IMP line. This sequence is synchronous on the line.

When no packets are being transmitted, the circuitry generates a continuous stream of SYN characters. When the program has a packet ready for transmission, it notifies the hardware which at the end of the current SYN character, indicates a DLE character and then a STX character. Note that at least two SYN characters are caused to precede a new packet sequence, i.e., the DLE and STX characters and the new packet.

The packet text must consist of an even number of 8-bit bytes. At the completion of the packet text, the circuitry appends another DLE character followed by an ETX character.



PACKETS: Multiples of 16 bits
Max. 1008 bits, not including control word.

TOTAL MESSAGE: 8096 bits max.; 9 packets 8 of which have msg. data.

FIGURE 4.10 - SEGMENTATION OF A MESSAGE INTO PACKETS FOR THE
VERY DISTANT HOST INTERFACE

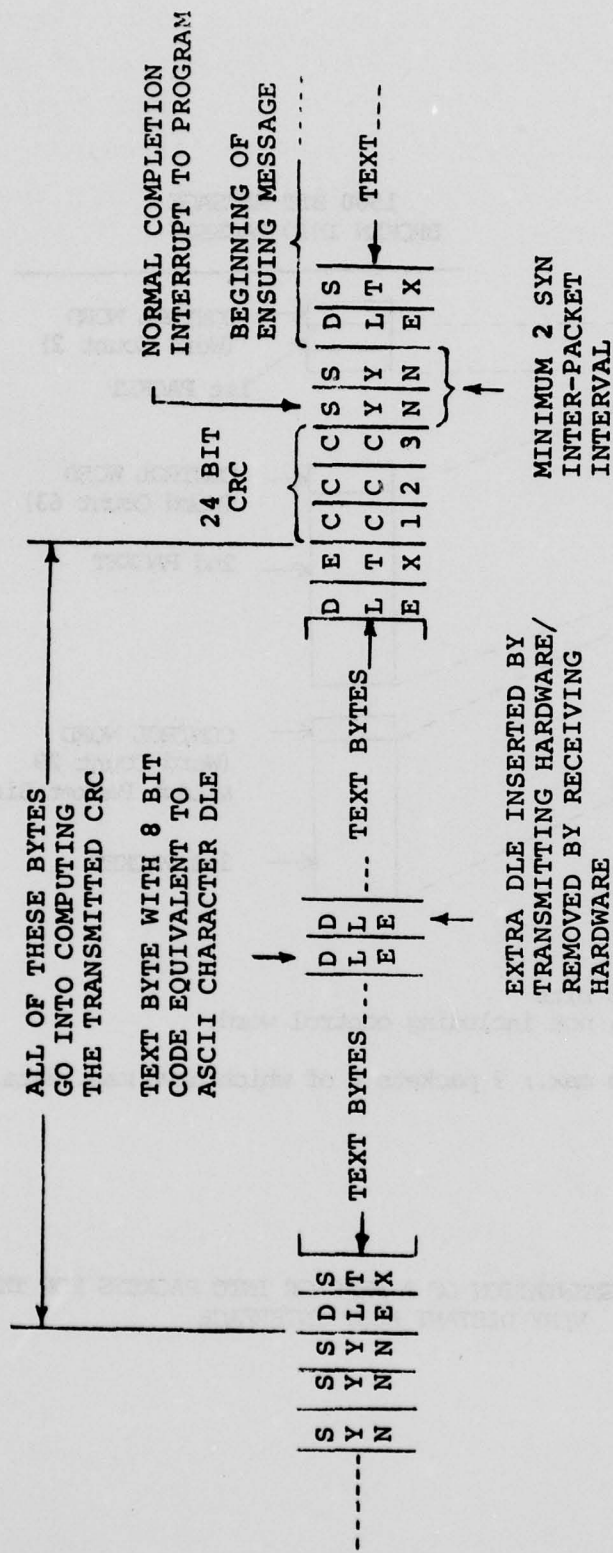


FIGURE 4.11 PACKET FORMAT ON LINE BETWEEN IMP AND VERY DISTANT HOST

A Special Packet is a one word packet consisting of the control word with the special packet bit set to zero. In a special packet, none of the control word fields or bits have their usual meaning; consequently a special packet cannot be used to acknowledge data packets or to send data. In the Special Packet, only two bits other than the Special Packet bit have any meaning - the Hello/I-Heard-You-Bit and the Host/IMP bit.

Every r seconds the IMP and Host independently transmit a Hello packet with Hello/I-Heard-You Bit set to zero. On receipt of a Hello packet the recipient must promptly return an I-Heard-You (Set to one) packet (highest priority) as an acknowledgement. If either IMP or Host sends more than t Hello packets without an I-Heard-You packet acknowledgement, the respective IMP or Host declares the line dead.

Once the line is declared dead the declarer must accept no packets (either special or regular) for $2 * t * r$ seconds to allow the other party to also declare the line dead. At the indicated wait an attempt is made to bring the line alive. This is done by sending Hello packets (but no regular packets) every r seconds while listening for I-Heard-You packet responses. It is necessary to have k Hello packets responded to in a row. If K responses are obtained in intervals of r , the line is declared alive, and regular packets can again be sent.

IMP restrictions require that transmissions to and from a very distant Host be in packets which are multiples of 16 bits in length, up to a maximum of 1008 bits. This does not include a 16 bits control word appended to the front of each packet. See Figure 4.10.

* At present r is 1.25 seconds, t is 4, and k is 4. These values are adjustable.

Note that if a DLE character occurs in the text of the packet, the circuitry adds a second DLE character. This insures that a DLE/ETX sequence cannot occur in the packet text as transmitted. The receiving circuitry, of course, also watches for the DLE and if one is detected, it deletes the accompanying DLE character added before transmission.

4.1.9.2.1 CYCLIC REDUNDANCY CHECK

Referring again to Figure 4.11, a 24 bit cyclic redundancy check is computed on the transmitted packet. As shown in the figure, the DLE and STX characters preceding the packet and the DLE and ETX characters following the packet are included in the computation.

The 24 bit CRC result is transmitted as three 8 bit bytes. Then at least two SYN characters are transmitted before the next packet sequence is allowed to commence.

4.1.0 REFER TO APPENDICES A AND B

For detailed descriptions of the leader formats of the ARPANET distant host interface and corresponding ADPT testbed leader interface. It is noted that the ADPT testbed requires those fields in the packet header (also described in Appendix B) which are not specific leader fields are to contain blank or zeroes. Furthermore a 32 bit FCS code (part of ADCCP) follows the data characters of each segment (not shown in the Appendix B).

AD-A049 619

RCA CORP CAMDEN N J

F/G 17/2

STUDIES FOR DEVELOPMENT OF A UNIFIED NODE EMPLOYING DYNAMIC CHA--ETC(U)

DEC 77 B PATRUSKY, K BODZIOCH, R CHAN

F30602-75-C-0109

RADC-TR-77-380

NL

UNCLASSIFIED

305
AD
A049619



REFERENCES

1. Host/Host protocol for the ARPA Network, ARPA Network Information Center, #7147.
2. TELNET protocol, T.O. Sullivan et al, ARPA Network Working Group Request for Comments #158, ARPA Network Information Center, #6768, May 1971.
3. Official Initial Connection Protocol, J.B. Postel, ARPA Network Working Group Document #2, Network Information Center #7101, June 1971.
4. The data transfer protocol, A. K. Bhushan et al, REC #264, Network Information Center #7812, November 1971.
5. The File Transfer Protocol, A. K. Bhushan et al, RFC #265, Network Information Center #7813, November 1971.
6. User's guide to the terminal IMP, Bolt Beranek and Newman Inc., Report No. 2183.
7. The BBN terminal interface message processor, BBN Report 2184.
8. Specifications for the Interconnection of a Host and an IMP, Report No. 1822, December 1974 revision.
9. Function - oriented Protocols for the ARPA Computer Network, S. D. Crocker, et al, Spring Joint Computer Conference, 1972.
10. Host-Host Communication Protocol in the ARPA Network, C.S. Carr et al, Spring Joint Computer Conference, 1970.

4.2 AUTODIN INTERFACE

The ADPT node prior to the present program included a message switch capable of interfacing mode I and mode II terminals. This message switch was tested as an Autodin interface in the previous phase of contract F30602-69-C-0050. The testbed was shown to be capable of appearing as DSTE type subscriber(s) to interface an Autodin access line. The ADPT program completely retains the original message switch hardware and software package including the mode I and mode II protocol line handlers and message exchange capability.

The ADPT phase, however, has incorporated new data terminals which interface the packet switch function of the testbed. It therefore became necessary to provide a mechanism for providing for transaction interchanges between the message switch and the packet switch terminals. This mechanism termed a "pseudo host" consists of a software package which can accept or deliver Autodin type message linked line blocks to or from the resident message switch function and provide for conversion to segments which interface the packet network. A small expansion of the message switch address directory was also necessary to include the packet switch terminals within the framework of the Janap 128 message header required by the message switch function.

Figure 4.12 indicates the message/packet interface. It is noted that Mode I and Mode II terminals are connected to the message switch and that message transfers between local terminals or between these terminals and remote Autodin terminals are locally handled wholly by the message switch function (as before).

The packet switch terminals (eg VDU's and MTC's) added in this phase are made interoperable with local (or remote) mode I or mode II terminals via the pseudo host.

MODE I/MODE II LINES

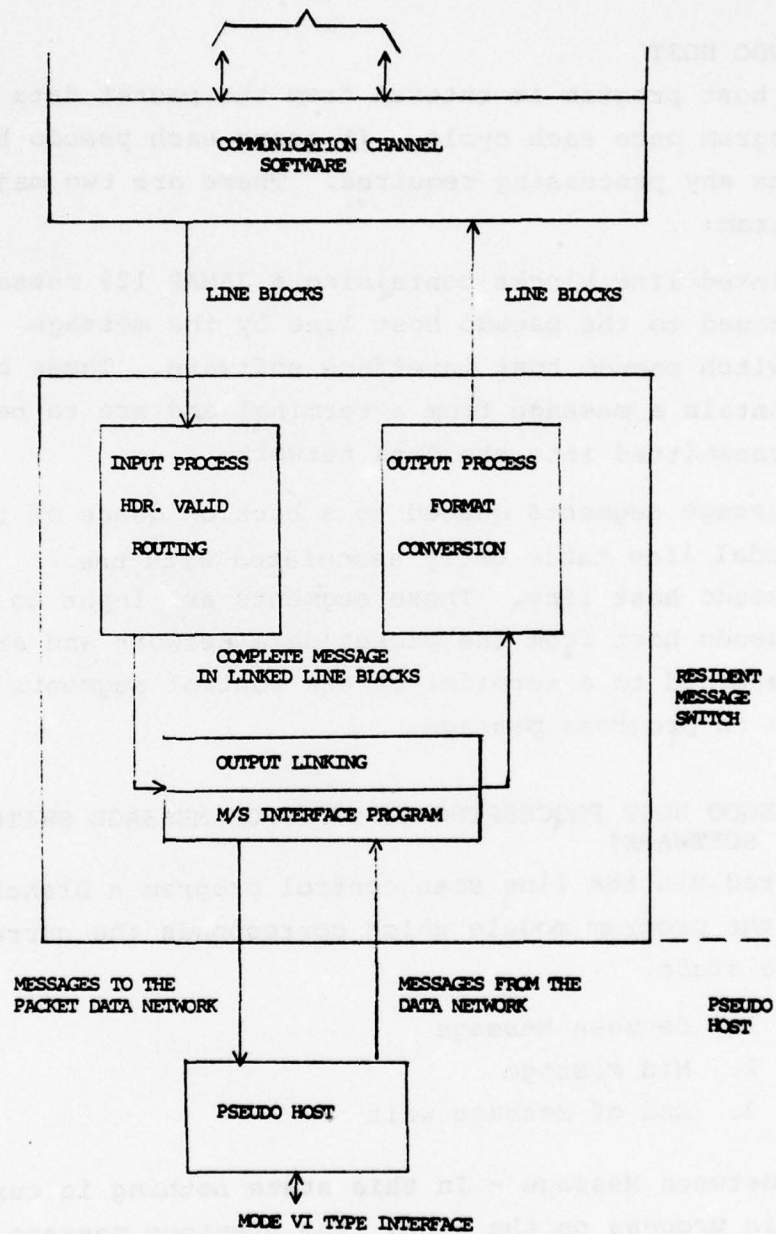


FIGURE 4.12 - MESSAGE/PACKET INTERFACE

The following paragraphs describe the pseudo host implementation, and the resident message switch capabilities.

4.2.1 PSEUDO HOST

The pseudo host program is entered from the packet data switch control program once each cycle. It scans each pseudo host line and performs any processing required. There are two major inputs to the program:

- a. Linked line blocks containing a JANAP 128 message queued to the pseudo host line by the message switch pseudo host interface software. These blocks contain a message from a terminal and are to be transmitted into the data network.
- b. Message segments queued to a backlog queue of the nodal line table entry associated with the pseudo host line. These segments are input to the pseudo host from the packet data network and are destined to a terminal or are control segments for an in progress message.

4.2.1.1 PSEUDO HOST PROCESSING INPUT (FROM MESSAGE SWITCH SOFTWARE)

When entered via the line scan control program a branch is taken to the program module which corresponds the current input line state.

1. Between Message
 2. Mid Message
 3. End of message wait
- a. Between Message - In this state nothing is currently in process on the line. The previous message has cleared and no new message has started. The program checks the input line block queue (ILBQ) to see if a message (series of

line blocks) has been linked up by the message switch interface program. If no message is received, no action is required. If a message has been received this unit will process that message. The originating and destination RI's are located and used to determine the originator and destination node and nodal line number. This is done by searching the pseudo host routing table for the occurrence of an RI (or part of an RI). The originator node/line number is used to locate the proper pseudo host line table. Various segment protocol header data is then stored in the pseudo host line table, and an SOM segment is placed in the new segment queue of the associated Nodal Line Table. Depending on whether or not all line blocks were used the line state is changed to mid-message - (more line blocks to process) or end of message wait - (all line blocks processed). In this last case the segment would indicate in the command code that segment is a single segment message (SOM/EOM).

- b. Mid Message - In this state, the program is waiting for the SNSG (send next segment) bit of the host line table to be set. The pseudo host output processing sets SNSG when a SNSG segment is received from the packet network. If SNSG is not set no processing takes place. When it is set then another segment is built and transmitted using line blocks in the ILBQ. The state is set to mid message or end of message wait depending on whether or not line blocks remain.
- c. End of message wait - All line blocks have been used up and an EOM segment has been sent. The program checks the RFNM bit to see if an RFNM control segment has been received. If so the state is set to between message and the pseudo host line entry reset to initial conditions.

4.2.1.2 PSEUDO HOST PROCESSING - OUTPUT

When entered by the line scan control program the backlog queues of the associated nodal line table are checked to see if any segment has arrived from the network. If so the segment is un-

queued and the proper program module entered depending on whether the segment is a data or control segment.

- a. Control Segment Processing - The control segment will be either an RFNM or SNSG. The proper bit (RFNM or SNSG) is set in the pseudo host line table and the segment released. The RFNM or SNSG will be checked and reset by the pseudo host input processing module.
- b. Data Segment Processing - The data segment is checked to see if it is an EOM segment in which case EOMR is set in the pseudo host line table. The segment is broken down into line blocks by removing the segment protocol header and filling all remaining data into line blocks. This assumes that the message is a proper JANAP 128 format when composed at a host. The new line blocks are then linked to the output line block queue (OLBQ) and the segment placed in the released segment queue of the associated ICC nodal line table entry. If EOMR is set then the output linking subroutine of the Message Switch Program is called to initiate transmission of the message by the Message Switch Output program.

4.2.1.3 INTERFACE TO MESSAGE SWITCH

Figure 4.12 showed an overview of the ICMS Message Switch and the interface to the Pseudo Host. Data from terminals is processed by the Communication Channel (CCSVC) software and passed to the Message Switch in line block form. These line blocks are collected by input processing and validated. Once a complete message has been received (it is now a series of linked line blocks) it is linked to the output process. It is in this output linking function that the Message Switch interface to the Pseudo Host has been made.

All messages input to the Message Switch are checked by the message switch interface program to see if they are destined for a terminal in the packet data network. If so the linked line blocks are

queued up to the Pseudo Host ILBQ. All other messages are allowed to go through, being linked up to the message switch output process.

Messages originating at a terminal in the packet data network and destined to a terminal (or Autodin access line) connected to the Message Switch are processed as follows. The message is changed from segment form to linked line blocks by the Pseudo Host. When all segments of the message have been converted the line blocks are linked to the Message Switch output process for transmission.

Figure 4.13 shows the current configuration of the testbed message switch used to check the pseudo host. The single node message switch services two Mode II teletypes (silent 700's) with two Mode I devices (card) tied back to back*. The interface to the Data Switch can be either Mode II or Mode I. Mode I input to the Message Switch is produced by the routing Mode II terminal input to a Mode I output. The message Switch will convert the message on output and will then receive the message as Mode I input since the lines are back to back. This Mode I input can then be passed to the Data Switch. For traffic from the Data Switch the reverse takes place.

The conversion of Mode II input to Mode I input is initiated by key characters in the language media/format fields of the JANAP 128 header. Traffic coming from the Data Switch is forced through the reverse process (Mode I output to Mode II output) based on the logical channel of the message. Even logical channels are transmitted to a Mode I device but routed to a Mode II device. Odd logical channels are routed and transmitted to the Mode II device. A Data Switch subscriber addresses a message using destination node/line numbers. Destination lines corresponding to pseudo hosts will have two devices (card and TTY) associated with them. The logical channel of the message determines which device.

* Since DSTE's were not available, the Mode I interface is simulated via the Mode II/Mode I message exchange software.

messages originating at a terminal in the packet data network and destined to a terminal (or terminals across links) connected to the message switch are processed as follows. The message is changed into packet form by the packetizer. The packets are then sent to the message switch. The message switch processes the packets and sends them to the destination terminal. The message switch also provides for transmission of data to the message switch output process for transmission.

Figure 4.13 shows the current configuration of the message switch used to check the packetizer. The message switch consists of two cards, CARD 1 and CARD 2, which are connected to a central MESSAGE SWITCH NODE. The MESSAGE SWITCH NODE is connected to two terminals, TTY1 and TTY2, via MODE II connections. The MESSAGE SWITCH NODE also receives MODE I TRAFFIC and sends MODE II TRAFFIC. The MESSAGE SWITCH NODE is connected to a DATA SWITCH via a MODE I connection.

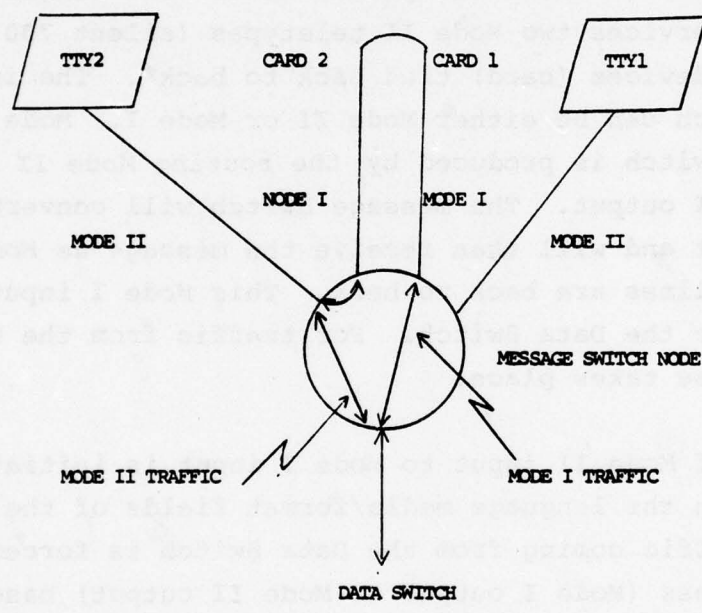


FIGURE 4.13 - LOGICAL CONFIGURATION OF MESSAGE SWITCH IN TESTBED

For messages originating at the Message Switch the last letter of the routing indicator will be used to determine logical channel. See terminal routing for details. (4.2.1.4)

4.2.1.4 TERMINAL ROUTING

As indicated in the Input "Between Message" state, new messages will be in JANAP 128 format. Figure 4.14 shows the JANAP 128 paper tape header format. This format will be used for all terminal inputs to the packet data network. The pseudo host must prepare the segment leader. Included in this leader is the node and line number of the called party. This is determined by the Addressee in the message header. (Character position 36 through 42). For the testbed, the following simplified routing plan was used:

Char#	36 -	37 -	38 -	39 -	40 -	41 -	42 -	43
	R	U	W	J	A	A	A	

Constant

Constant

Routed on this
if other valid

Positions 36 through 41 must match the constant RUWJAA and 43 must have a period. (End of Routing Signal) If valid, position 42 is looked up in the pseudo host routing table to get node and line number of the addressee. If one of the constants is wrong or the character in position 42 is not in the table, the message is rejected. Character 42 determines host logical channel in addition to line routing. Every alternate alphabetic letter starting with A will route to logical channel one. Every alternate letter starting with B will route to logical channel four.

4.2.2

The resident message switch program will be briefly described based on information extracted from the ICMS final report.

d. Paper Tape Header Format

Leader (para. 422c(1))	_____	
Precedence	_____	
Language Media and Format	_____	
Classification, as appropriate	_____	
Separator	_____	
Content Indicator	_____	
Separator	_____	
Originator	_____	
Station Serial Number	_____	
Separator	_____	
Julian Date	_____	
Time Filed	_____	
Classification Redundancy	_____	
Start of Routing Signal	_____	
Addressee	_____	
End of Routing Signal	_____	

RT U J A M R U W T A A A 1 2 3 4 2 2 0 1 9 1 5 - U U U U - - R U W J A A A

FIGURE 4.14 JANAP 128 HEADER

The message switching software is divided into two distinct areas, namely Line Management Programs, which function only in State 2, and Message Processing Programs, which generally function in State 1.

The Line Management Programs are designed to be time-independent of the Message Processing Programs with the exception of waits for the Processing Programs to take specific actions which are not time-critical.

The Message Processing Programs are designed on a demand basis, with the Executive Program deciding what processing is to be performed next. The Executive Program provides the enabling interface communications between the Line Management Programs and the Processing Programs. Upon completing the successful framing of a line block of message data on Input, a channel flag is raised by the Line Management Routing. This is recognized by the Executive Program which passes control to the Input Processing Routine to process the data into the system's intransit store. When a message is ready to send out, the Processing Program raises a flag which is recognized by the Executive, which will activate the Output Line Management Routines to request data for transmission.

The Executive then passes control to the Output Processing Programs to process the message data into the Line Management buffer area for transmission on the line.

By this technique, the Message Processing Programs may be spending all their time input message processing, and not entering the output processing programs at all if there is nothing to do on output, or vice-versa.

The Message Switch Unit (MSU) controlled devices are synchronous and asynchronous character buffers connected to MSU subscriber terminals. The buffers are serviced in accordance with the requirements specified in DCA 370-D175-1 for Mode I and Mode II

channels. The servicing routines are segregated by channel type, direction of traffic (input or output), and device status. The front-end channel number used to access channel status information is determined from the Communications Channel Device Status Table.

4.2.2.1 EXECUTIVE PROGRAM

An Executive Program controls all State 1 Message Processing operations, all input-output operations to peripheral devices, and all State 1 interrupts.

The Executive Program consists of the following sections:

- Channel Service Scans

- Channel Timers

(a) Executive Channel Service Scans

These routines are entered by the Executive Control when message processing can be done if there is anything to process. There is a series of Channel Scan Words which are examined by the Executive Scan, to determine the processing to be performed. The priority of processing is determined by which scan words the Executive examines first and in which order they are processed.

Each Scan Word contains one unique bit position for each channel which is set by the appropriate routines when the channel requires the particular service represented by the scan word. Thus, the Executive Scan goes to the proper processing routine represented by the scan word, with the channel number predetermined by the bit number having been set. Before transferring control, the Executive Scan resets the channel bit (this operation must be non-interruptable by the Line Management program interrupts).

A scan word is required for each of the following functions:

1) Input Service Scan Word (ISS)

The bit for a channel is set by the Line Management (LM) Program whenever a line block has been completed for the channel, or the LM Program has set an Action Command

Indicator (ACI). The Executive will transfer program control to the Input Processing Programs to process the line blocks available or to take action according to the ACI.

2) Output Service Scan Word (OSS)

The bit for a channel is set by the Input Processing Program whenever a message is linked and released to output and the message is the only one on the channel link.

The bit is also set by the Line Management output routines upon every successful transmission of a line block, and indicates it is ready for the next line block. The bit being set will cause the Executive to enter the Output Processing Program to process the next line block and for the LM Program to send out on the line.

(b) Channel Timers

The central controller has a program-settable elapsed time counter which can cause a State 1 interrupt when it has decremented to zero. The program to which control is transferred on interrupt is considered as part of the Executive. Each timing function of the system, including both Line Management and Message Processing Programs, have an associated counter and a channel indicator word which define those channels keeping an active timer.

4.2.2.2 MESSAGE LINKAGE FROM INPUT LINE TO OUTPUT LINE

The method of linking messages throughout the system for the model is based on the fact that the intransit message store of the system is located in high speed memory of the controller. The tables and address pointers and their general usage, are described below.

(a) Data Buffer

A linked buffer area, each entry of which contains space for 84 message line block characters, an address field containing the

address of the next line block of the chain, and certain other control indicators. A vacancy chain is maintained in the Data Buffer with vacant links being seized by the Line Management Programs as required and filled with incoming data characters, and the Output Processing Program re-linking line blocks of messages which have completed transmission into the vacancy chain.

(b) Output Buffer

A linked buffer with the same entry format as the DATA BUFFER, and having its own vacancy chain. A vacant entry is seized by the Output Processing Program when it must process the next line block to transmit for a channel. The Output Processing Program processes message data from its current Data Buffer line block into the Output Buffer line block until the Output Buffer line block has been filled. Output Processing then links the Output Buffer line block over to the Line Management Program for transmission. The Output Buffer entries are unlinked and re-linked to the vacancy chain by the Line Management Program when the line block has been successfully transmitted.

(c) Front End Link Buffer (FELB)

A linked buffer area with one vacancy chain, each entry containing two address fields. One address field points at the next FELB entry in the active chain; if one is present, the other address points at the first line block of the associated message in the Data Buffer. One FELB entry is always linked to each Line Management Program input channel. Upon receiving a start-of-message character, the Line Management Program seizes a vacant line block in the Data Buffer and links it to the input channel. As characters are received, they are stored in the Data Buffer Line Block. When the block is full, another Data Buffer line block is seized and used for character storage. Upon reaching the EOM (End of Message) of the message, the Line Management Program seizes an entry from the FELB, to be used as the start-of-message pointer for the next message.

(d) Message Link Table (MLT)

A linked buffer area with one vacancy chain, each entry containing message classification and identification information and two address fields; one containing the address of the SOM (Start of Message) line of the message in the Data Buffer, and one containing the address of the next MTL entry in the chain.

When the Input Line Management Program completes an input line block, it sets an indicator in the Data Buffer line block entry if it has completed this line block and sets the channel bit in the Executive Program's Input Service Scan Word.

The Input Processing Program is then entered and locates and processes the completed line block. Upon processing the EOM, the Input Processing Program unlinks the FELB entry and builds an MLT entries for each transmission of the message. These MLT entries are linked to the output channels for transmission.

The Output Processing Program seizes a vacant line block from the Output Buffer vacancy chain, and fills it with message data from the message link line blocks of the data buffer. When an Output Buffer line block is filled, the Output Processing Program links the line block over the Line Management output channel, and sets the proper indicators to cause the Line Management Program to start sending out the processed line block.

When the line block has been successfully transmitted, the Line Management Program relinks the Output Buffer line block back to the vacancy chain.

As the Output Processing Program completes transmission of each copy of a message, it unlinks the MLT entry and relinks it to the MLT vacancy chain. When a message has had all copies transmitted, then the message data line blocks in the Data Buffer are unlinked and relinked to the Data Buffer Vacancy Chain by the Output Processing Program.

4.2.2.3 LINE MANAGEMENT - PROCESSING PROGRAM INTERFACE

The program interface between the Line Management Programs and Processing Programs is primarily accomplished through the Executive Program Scan Words and associated Action Command Indicator or Channel Action Indicator. In addition, there are other minor indicator interfaces and/or expected actions or reactions which must occur in order to obtain proper operation. These are detailed in the following paragraphs.

(a) Input Interface

As the Line Management Program is filling line blocks with characters, the Input Processing Program is (or could be) processing previously completed line blocks. Upon completing a line block, the Line Management Program sets an indicator in the control portion of the Data Buffer line block and the Input Processing Program can start processing. This will be true regardless of mode of transmission. Also, the Input Service Scan Word of the Executive Program is set upon the Line Management Program completing a line block. The Input Processing Program will be attempting to process all line blocks available to it when it is entered by the Executive. The possibility exists that Line Management may complete a line block for the channel which Input Processing is currently processing, and Input Processing would complete processing this new line block before returning control to Executive. In this event, the Executive would detect the Scan Word still set for the channel, and go back to Input Processing which would sense that there was nothing to do and exit back to the Executive.

(b) Output Interface

Upon being entered to start processing a message for an output line, the Output Processing Program seizes a vacant line block from the Output Buffer chain of the Data Buffer and processes into it the first portion of the message to be sent out. In order to chain this line block to the Line Management Program, the Output Processing Program first sets a Line Management table indicator. Output

Processing then determines if the output block count is zero. If zero, the relative address of the first line block being chained is stored, and the relative address of the last line block of the chain being linked is stored. The number of line blocks being linked to the channel is added to the output block count, the ACTIVATE CHANNEL bit is set, and the Line Management table indicator is reset.

If the output block count isn't zero, the Output Processing Program loads the relative address of the first line block being linked and stores the relative address of the last line block of the chain being linked. The number of line blocks being linked to the channel is added to the block count, the ACTIVATE CHANNEL bit is set, and the LM indicator bit is reset. Upon completing successful transmission of the line block, the Line Management Program relinks the line block back into the Output Buffer vacancy chain, decrements the block count by one, and sets the Executive scan word to re-enter Output Processing for the next line block.

After processing an EOM line block and linking it to the LM Program for transmission, the Output Processing Program can determine successful transmission of a message by detecting that the output block count has gone to zero.

4.2.2.4 INPUT PROCESSING

The registers are initialized and housekeeping functions are performed. A check is made to determine if one of the Action Command Indicators (ACI) is set. If so, control is transferred to the ACI sub-routine. Next a check is made to see if an RM (Reject Message) has been requested on this channel. If so, transfer is made to the RM Routine. If not it must be determined if data is available for this channel. This is done by checking the first entry for this channel in the Front End Link Buffer. If data is not present, control is returned to the Executive Routine. If present, a check is made to see if this line block is available for input processing.

This is done by checking the status in the Data Buffer Entry addressed by data. If it is not available, transfer is made to the Executive Routine.

If data is present and available the Channel Type is checked in the Channel Utilization table. If Mode II, control is transferred to Mode II Input Processing. If Mode I and End of Message (EOM) is set in this line block, Input Channel Status is checked to see if this is also Start of Message (SOM), and if so, SOM processing is initiated. If the EOM indicator is not set, status is checked to see if this is SOM processing. If not SOM this is a mid-message line block. The address of the Last Data Buffer Entry Processed (LDA) is updated. Then, if this is a card message, the running card count is updated. A check is made to see if there are more line blocks available. If so, transfer is made to check next line block. Otherwise control is transferred to the executive routine.

(a) SOM Processing

The first function of SOM processing is to enter the Header Validation Subroutine. This routine validates the invariant header of the message through Start-Of-Routing. If any discrepancy is found, the message is rejected and an appropriate printout is issued and control returned to the Executive Routine. Otherwise control is returned to Input Processing.

Upon return from the Header Validation Subroutine the Routing Indicator (RC) Processing Subroutine is entered. This routine validates the routing indicators and builds a link in the message link table for each Routing Indicator found in the message. If an invalid routing indicator(s) or invalid format is found, the message is rejected, an appropriate printout issued, and control returned to the Executive Routine. Otherwise control is returned to Input Processing.

(b) EOM Processing

The Channel Utilization Table is checked to determine if the current assignment is a card message. If so, the card count from the

message is checked against the accumulated card count and, if there is a discrepancy, a printout is issued, the message is rejected, and control is returned to the Executive Routine. If the card count is correct or if this is a teletype message, the Link-Output Subroutine is entered. This routine removes the link(s) from the input channel chain and links them to the appropriate Output Channel Chain(s). Control is then transferred to the Input Processing Routine. An indicator in the Channel Status Table is set to SOM for future servicing and the Last Data Entry Processed indicator is set to indicate the next line block to be processed. The EOM ACK indicator is next set up for the Front End Program, and this channel's bit is set in the appropriate Scan Word. The address of next Front End Link Buffer Entry is moved, and the Front End Link Buffer Entry is now linked back to the end of the Vacancy chain and control is transferred to the Executive Routine.

4.2.2.5 OUTPUT PROCESSING

The Output Processing Program is entered only when the Executive Program determines that an output function is to be executed.

The number of the output channel to be processed is stored in a register before entry into the output program. The channel number is used to address the tables that are necessary to determine the type of processing required and also to locate the data to be processed.

(a) Reject Message (RM) Processing

The first step in output is to test the Action Command Indicator (ACI) for RM received. If the RM received is set, this indicates that the last message being transmitted over this output channel has been rejected. Output processing will reset the indicators and set the channel code to retransmit the message.

(b) Discard Message (DM) Processing

If the DM bit is set for the output channel, this indicates that the message is to be discarded by output. If the output channel is Mode I, the command from output processing to line management (CAI)* will be set. This will instruct the line management program to send a CAN* over the output channel.

If the DM bit is set and the output channel is Mode II, output processing will generate and send a Cancel Transmission Message (CANTRAN) to the output channel.

(c) Channel Status Processing

To determine the status of the channel and message being transmitted, Output Processing interrogates the channel status code.

- (1) If the setting is zero (0), the channel is out of service and the output program returns to the Executive Program.
- (2) If the setting is one (1), this indicates a start of message (SOM). At SOM time, a test is made to determine if there is any data to be processed. If there is not data to be processed for this output channel, the Output Program returns to the Executive Program. If there is data to be processed, the busy bit for the output channel is set. The input Message Format (IMF) and the Preferred Output Message Format (POMF) settings in the Message Link table are then tested to determine how to set the type of Output Processing in the Channel Table. If the POMF is not available on the output channel, the acceptable output type of equipment available must be used.

At SOM time if the output type of format is Mode II ITA#2, a preamble will be generated in the output buffer and the setting of the channel status will be changed. Output processing will then return to the Executive Routine.

*"CANCEL" is denoted by "CAN" and "CHANNEL ADDRESS INDICATOR" is denoted by "CAI."

- (3) If the setting of the channel status is two (2), this indicates mid-message processing.
- (4) If the setting is three (3), it indicates a EOM ACK. The output block count is tested and if not zero indicates that the Line Management Program has not finished processing the output channel and the Output Processing Program will return to the Executive Program.

If the block count is zero, the status Code is set to SOM (1) and the number of copies remaining to transmit indicator will be decremented by one. If this indicator is zero, the relative address of the message in the data buffer is moved into the last lineblock in the vacancy chain. This will link this lineblock to the vacancy chain. The relative address of the last lineblock for this message is moved into the Vacancy Status List. This will be the relative address of the last vacant lineblock in the data buffer. Output processing then returns to the Executive Program.

4.2.2.6 MESSAGE EXCHANGE

When Message Exchange is entered, the data to be processed will be in the data buffer and it will be necessary to move or convert this data into the Output Buffer.

To determine the type of Message Exchange required for this transmission, the type code in the Channel Table is tested.

The setting of type code (OPTYPE) can be:

- 0 - Card to Card
- 1 - Card to Mode I TTY
- 3 - Mode I TTY* to Card
- 4 - Mode I TTY* to Mode II TTY
- 5 - Mode II TTY to Card
- 6 - Mode II TTY same mode
- 7 - TTY to TTY same mode

* This refers to the paper tape reader on the DSTE.

If OPTYPE is Card to Card (0), the lineblock including the framing characters can be moved directly from the data buffer into the output buffer.

If OPTYPE is Card to Mode I (1) or Card to Mode II (2), the data conversion consists of inserting the correct teletype machine function characters into the output buffer lineblock and maintaining the lineblock size of eighty data characters in the output buffer.

If OPTYPE is Mode I to Card (3) the data conversion consists of deleting the machine function characters and starting a new lineblock on the recognition of the eight-first data character or a linefeed character, whichever comes first.

If OPTYPE is Mode I to Mode II (4) the data conversion consists of inserting any additional machine function character that are required, and maintaining lineblock size of eighty data characters in the output buffer.

If OPTYPE is Mode II to Card (5) the conversion consists of deleting all machine functions and maintaining lineblock size either eighty data characters or by ending a line when a carriage return linefeed sequence is recognized.

If OPTYPE is Mode II to Mode I (6) the conversion consists of deleting the machine function characters not required for Mode I teletype.*

If OPTYPE is Mode I to Mode I (7) or Mode II to Mode II (7) the data characters of the lineblock in the data buffer are moved directly into the lineblock in the output buffer.

When a Message Exchange is required, especially card to teletype, the input lineblock may expand to greater than eighty data characters because of the insertion of machine function characters.

This may necessitate the rereading of a lineblock to pick up the characters that would not fit in the previous output lineblock. To determine which input data characters are to be processed first, the relative address of the next data character to be processed is maintained by the message exchange routine.

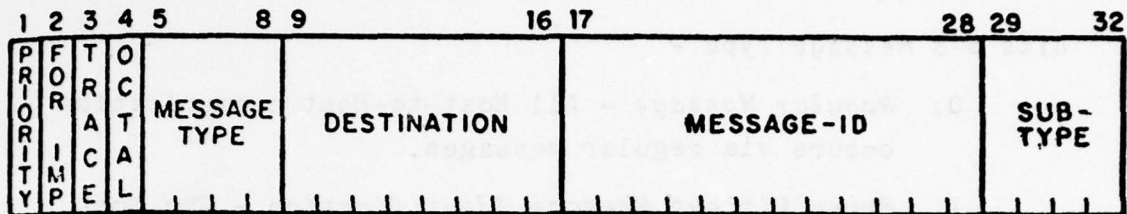
When message exchange compresses a lineblock such as teletype to card conversion, where the teletype machine functions are deleted, it may be required to process more than one lineblock to obtain one output lineblock. When a message is ready to be processed by the Output Program the entire message will be in the data buffer.

For each lineblock that is processed, the Output Program must inform the Front End Program that a lineblock is ready for transmission and the location of the lineblock in the Output Buffer. This is accomplished by setting the Front End Table with the pertinent information required. Upon completion of this procedure the Output Program returns control to the Executive Program.

* This refers to the paper tape reader on the DSTE.

ARPANET HOST - IMP LEADER FORMAT

Host-to-IMP Leader Format



HOST-TO-IMP LEADER FORMAT

Bit 1 Priority -

Most messages should have this bit set to zero; messages with this bit set to one will be treated as priority messages (see Section 3.1).

Bit 2 For IMP -

The For IMP bit, which is designated for debugging, changing IMP parameters, IMP Teletype output, and discarding packets, is discussed in Section 5. The Host should normally set this bit to zero. In particular, the Host should not set this bit on control messages to his own IMP.

Bit 3 Trace -

If equal to one, the message is designated for tracing as it proceeds through the network so that reports on this message's transit through the network may be sent to a trace destination. (The tracing process is described in Section 5.)

Bit 4 Octal -

This bit applies only to messages directed to the IMP Teletype. If equal to one, the message will be printed on the Teletype as a sequence of octal

numbers, each representing one 16-bit IMP word. If equal to zero, the message will be printed as a sequence of ASCII characters.*

Bits 5-8 Message Type -

0. *Regular Message* - All Host-to-Host communication occurs via regular messages.
1. *Error Without Message Identification* - The Host program detected an error in a previous IMP-to-Host message and had to assume that the leader was garbled; i.e., the leader had a format error, or the Error flip-flop was set during transmission of the leader.
2. *Host Going Down* - It is assumed that as the time for the Host to (voluntarily) go down approaches, the Host itself will send warning messages to its network users. Just before going down, the Host should send the Host-Going-Down message to its IMP. The Host should then (if it can) continue to accept messages from the IMP for a period of 5 or 10 seconds, to allow messages already in the network to reach it. The IMP will store the Host-Going-Down message and return it to any source Host along with Destination (Host) Dead messages. The IMP will try to preserve this message over IMP reloads where appropriate. The NCC will be able to manually update the stored copy of this message in response to a phone call from the Host site in the event the Host is going to be down longer than it said or if it did not have time to give warning before going down.

*The IMP's internal ASCII character set is listed in Appendix E.

Bits 17-28 (the message-id field) of the Host-Going-Down message give the time of the Host's coming back up, bit-coded as follows:

Bits 17-19: the day of the week the Host is coming back up. Monday is day 0 and Sunday is day 6.

Bits 20-24: the hour of the day, from hour 0 to hour 23, that the Host is coming back up.

Bits 25-28: the five minute interval, from 0 to 11, in the hour that the Host is coming back up.

All three of the above are to be specified in Universal Time (i.e., G.M.T.). The Host may indicate that it will be coming back up more than a week away by setting bits 17-28 all to ones.

Setting all bits 17-27 to one and bit 28 to zero means it is unknown when the Host is coming back up.

Bits 29-32 (the sub-type field) of the Host-Going-Down message should be used by the Host to specify the reason it is going down. These bits are coded as follows:

<u>Value</u>	<u>Meaning</u>
0-4	Reserved for IMP use
5	Scheduled P.M.
6	Scheduled Hardware Work
7	Scheduled Software Work
8	Emergency Restart
9	Power Outage
10	Software Breakpoint
11	Hardware Failure
12	Not scheduled up
13-15	Currently Unused

3. *Uncontrolled Packet* - A Host-to-Host packet for which the IMPs perform no message-control functions. Use of this message type is described in Section 3.7.
4. *NOP* - The IMP will discard this message, which is intended for use during initialization of IMP/Host communication. A simple rule for the Host to follow is to send a few *NOP* messages whenever the Host or the IMP has been down either voluntarily or involuntarily.
5. *Unassigned.*
6. *Unassigned.*
7. *Unassigned.*
8. *Error with Message Identification* - The Host detected an error in a previous IMP-to-Host message after the leader was correctly received; e.g., the message was too long, or the IMP Error flip-flop was set after transmission of the first packet of a multiple packet message but before the end of the message. A message of this type will have a leader whose assigned bits are identical to the assigned bits in the leader of the message in error except that the message type bits will be changed to have value 8.
- 9-15. *Unassigned.*

Bits 9-16 Destination -

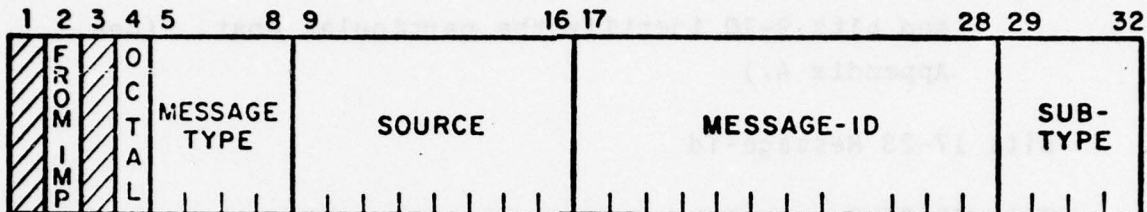
Identify a destination IMP and the particular Host at the IMP site. Bits 11-16 identify the destination IMP and bits 9-10 identify the particular Host. (See Appendix A.)

Bits 17-28 Message-id

Bits 29-32 Sub-type

The leader for all Host-to-IMP messages must be exactly 32 bits, even though some of these bits may be unused. The IMP will detect any message containing fewer than 32 bits (exclusive of hardware padding) and will return a Type 1 (sub-type 1) error message. The IMP will detect a message of any of the unassigned types (3, 5-7, 9-15) and return a Type 1 (sub-type 2) error message.

IMP-to-Host Leader Format



IMP-TO-HOST LEADER FORMAT

Bit 1 Unassigned.

Bit 2 From IMP -

If equal to a one, the message is from the Source IMP, rather than the Source Host. The From IMP bit is designated for debugging, IMP Teletype input, TRACE messages, and STATISTICS, and is discussed more fully in Section 5.

Bit 3 Unassigned.

Bit 4 Octal -

This bit applies only to messages directed to the IMP Teletype. If equal to one, the message will be printed on the Teletype as a sequence of octal numbers, each representing one 16-bit IMP word. If equal to zero, the message will be printed as a sequence of ASCII characters.*

*The IMP's internal ASCII character set is listed in Appendix E.

Bits 5-8 Message type

0. *Regular Message* - All Host-to-Host communication occurs via regular messages. The subtype field is unused.
1. *Error In Leader* - The IMP detected an error in a previous Host-to-IMP message and had to assume that the leader was garbled.

Sub-types:

0. IMP's Error flip-flop set during the first 32 bits of a message (see Section 3.2).
 1. IMP received a message of less than 32 bits.
 2. IMP received a message of an illegal Type.
2. *IMP Going Down* - The IMP will transmit this message to its Host before it voluntarily goes down. The Host should forward the information in the message to its users from the network (and to its own users of the network).

Bits 17-32 of the message are coded as follows:

Bits 17-18: Why;

0. "last warning" or "panic restart": the IMP is going down in 30 seconds.
1. Scheduled hardware PM
2. Scheduled software reload
3. Emergency restart

Bits 19-22: How Soon; in 5 minute increments
(zero implies immediately)

Bits 23-32: For How Long; in 5 minute increments
(zero implies immediately)

3. *Uncontrolled Packet* - A Host-to-Host packet for which the IMPs perform no message-control functions. Use of this message type is described in Section 3.7.
4. *NOP* - The Host should discard this message. It is used during initialization of IMP/Host communication. All fields except message type are unused.
5. *RFNM* - "Ready for Next Message". The named regular message was successfully delivered to the destination IMP, and the destination Host began to accept it. In addition, if the named message was longer than one packet (about 1000 bits) space is reserved at the destination IMP for another transmission, but the space reservation will remain valid for only a short time (see Section 3.1).

6. *Dead Host Status* -

Bits 17-28 (the message-id field) have the same meanings as bits 17-28 in the Host-to-IMP Type 2 (Host-Going-Down) message described in Section 3.3.

Bits 29-32 (the sub-type field) have the following meanings:

<u>Value</u>	<u>Meaning</u>
0	Currently Unused
1	The destination Host is not communicating with the network — it took its ready-line down without saying why.
2	The destination Host is not communicating with the network — the Host was tardy in taking traffic from the network without saying why.

<u>Value</u>	<u>Meaning</u>
3	The destination Host does not exist to the knowledge of the NCC.
4	The IMP software is preventing communication with this Host; this usually indicates IMP software re-initialization at the destination.
5	The destination Host is down for scheduled P.M.
6	The destination Host is down for scheduled hardware work.
7	The destination Host is down for scheduled software work.
8	The destination Host is down for emergency restart.
9	The destination Host is down because of power outage.
10	The destination Host is stopped at a software breakpoint.
11	The destination Host is down because of a hardware failure.
12	The destination Host is not scheduled to be up.
13-14	Currently Unused.
15	The destination Host is in the process of coming up.

When the value of the sub-type field is 1, 2, 3, 4, or 15, the message-id field will have the "unknown" indication.

Bit 1 in this message will always be set to zero and Hosts receiving this message should discard (without reporting an error) Type 6 messages

with bit 1 set to 1. This will allow the later addition of similar status information on dead destination IMPs.

The Dead Host Status message will be returned to the source Host shortly (immediately, if possible) after each Destination Host Dead (Type 7) message. The Destination Host Dead message applies to a specific named message, although the information contained in the Destination Host Dead message should probably be reported to all users connected to the dead Host. The Dead Host Status message does not apply to a specific named message and all users connected to the dead Host should be notified of the information contained in the Dead Host Status message.

Destination Host or IMP Dead (or unknown) -

This message is sent in response to a message for a destination which the IMP cannot reach. The message to the "dead" destination is discarded.

Sub-types:

0. The destination IMP cannot be reached.
1. The destination Host is not up.
2. Currently unused.
3. Communication with the destination Host is administratively prohibited.
- 4-15. Currently unused.

8. *Error in Data* - The IMP's Error flip-flop was set after transmission of the leader of a message but before the end of the message. The sub-type field is unused.
9. *Incomplete Transmission* - The transmission of the named message was incomplete for some reason. An incomplete transmission message is similar to a RFNM, but is a failure indication rather than a success indication.

Sub-types:

0. Destination Host did not accept the message quickly enough.
 1. Message was too long (in excess of 8095 bits).
 2. The message spent more than 15 sec. in transmission from the source Host to the IMP. This time is measured from the last bit of the leader through the last bit of the message.
 3. Message lost in the network due to IMP or circuit failures.
 4. Resources unavailable (see Section 3.1).
 5. Source IMP I/O failure during receipt of this message.
10. *Interface Reset* - The IMP's ready line has been dropped and pending output to the Host has been discarded (see Section 3.2). This probably indicates that the Host did not accept data from the IMP fast enough. Since dropping the ready line also sets the IMP's error flip-flop, the

next message from the Host will be discarded and answered with a Type 1 (sub-type 0) message. The sub-type field is unused.

11-15. *Unassigned.*

Bits 9-16 Source -

Identify a source IMP and the particular Host at that IMP site. Bits 11-16 identify the source IMP and bits 9-10 identify the particular Host. (See Appendix A:)

Bits 17-28 Message-id -

The message-id assigned to this message by the source Host.

Bits 29-32 Sub-type -

The sub-type field is used to qualify some message types as described above.

Messages of other than Type 0 are sent to the Host prior to messages of Type 0. The assigned bits in a leader for messages of Types 5, 7, 8, and 9 are identical to those in the received leader from the Host except that the message type and sub-type bits will be changed.

APPENDIX B

HOST INTERFACE LEADER/HEADER
FORMATS FOR ADPT TESTBED SIMULATOR

TABLE 1 - LEADER/HEADER

BIT #		FORMAT								SEG	PACK	SEG	
		1	2	3	4	5	6	7	8	HOST - NODE	NODE - NODE	NODE - HOST	
1	ADCCP	{	ADDRESS								Y	Y	Y
2			0	N(S)		P/F		N(R)		Y	Y	Y	
3			PRECEDENCE								Y	Y	Y
4			CATEGORY								Y	Y	Y
5			LOGICAL CHANNEL								Y	Y	Y
6			SECURITY								Y	Y	Y
7			COMMANDS								Y	Y	Y
8			DEST NODE								Y	Y	Y
9			DEST LINE								Y	Y	Y
10			SEQ NUMBER/AA								Y	Y	Y
11			ORIG NODE								N	Y	Y
12			ORIG LINE								N	Y	Y
13			IDA								N	Y	N
14			F _{OR} F _{RM} R _{FC} C _C V _{OICE} C _{RR} T _{RACE}								N	Y	N
15	NODAL LINE NO.								N	-	N		
16	BYTE COUNT								N	-	N		
17													

256

240 DATA CHARS MAX.

NOTES: BIT 1 = Low Order Bit
BIT 1 = First Bit Transmitted

BYTE FIELD DESCRIPTIONS

1. ADDRESS - A = HOST 1, B = HOST 2, C = R100 (ADCCP)

2. ADCCP CONTROL WORD (See Table 2)

3. PRECEDENCE

Z or Y = FLASH

O = OPERATIONAL IMMEDIATE

P = PRIORITY

R = ROUTINE

4. CATEGORY

*A = SUPERVISORY

B = INTERACTIVE or QUERY/RESPONSE

C = NARRATIVE/RECORD

D = BULK 1

E = BULK 2

*A is used only by network.

5. LOGICAL CHANNEL

ASC11 1 = VDT → VDT (OR TERM)

ASC11 2 = TAPE → TAPE (OR TERM)

ASC11 3 = TAPE (OR TERM) → VDT

ASC11 4 = VDT (OR TERM) → TAPE

NOTE 1: MEMO ADPT-T-0023

NOTE 2: LOGICAL NUMBER IS TREATED AS BINARY. LOGICAL NOS.
0-128 ARE POSSIBLE. PARITY BIT "B8" IS IGNORED.

6. SECURITY

ASC11 T = TOP SECRET
ASC11 S = SECRET
ASC11 C = CONFIDENTIAL
ASC11 E = ENCRYPTED FOR TRANSMISSION ONLY
ASC11 U = UNCLASSIFIED

7. COMMAND CODE DESIGNATORS - See Table 3.

8. DEST. NODES - ASC11 Numbers 1-9 are useable. Demo will use only the Codes resulting from ASC11 numerics 1, 2 and 3 (See Table 4).

Experiment may use more nodes; however, field will be restricted to ASC11 numerics 1 thru 9.

9. DESTINATION LINE NO's. - Demo/experiment will only use those codes resulting from ASC11 numerics 1 through 9.

10. SEQUENCE NO. - Segment sequence number is treated as a binary number. Starting value for VDT inputs is binary equivalent of ASC11 "1". Parity bit "B8" is ignored. Range = Binary 0 to Binary 128. Sequential segments must use sequential sequence numbers.

10A. AA - Sequence number acknowledged by destination node indicating node has delivered this block to the destination host (occupies same byte entry location in response as sequence no. in data sent and has same format).

11. ORIGINATOR NODE - Same format as destination node (see byte entry 8).

12. ORIG. LINE NO. - Same format as destination line no. (see byte entry 9).

13. IDA - In Delivery Acknowledgment Binary numeric indicating to source node highest sequence number block that can be accepted by destination node (Range 0-128).

14. NETWORK CONTROL BYTE (BIT INDICATORS ARE SET AS REQUIRED)

BIT 1 = FOR	MESSAGE IS FOR A NODE, NOT A LINE
BIT 2 = FRM	MESSAGE WAS GENERATED BY A NODE, NOT A LINE
BIT 3 = RFC	REQUEST FOR CONNECTION
BIT 4 = CC	CONNECTION CONFIRMED
BIT 5 = VOICE	INDICATES MESSAGE IS VOICE SIGNALLING PACKET
BIT 6 = ERR	INDICATES RCV BLOCK ERROR
BIT 7 = TRACE	USE IN DEMO TO INITIATE PRINTOUTS

15. NODAL LINE NO. - Byte location used by node software as scratch pad - Initial entry indicates Nodel Line No. (entered by interrupt program).

16. BYTE COUNT - Byte location used by node software as scratch-pad. Initial entry indicates number of bytes in received block (entered by interrupt program).

TABLE 2 - ADCCP CONTROL
FIELD FORMATS (BYTE #2)

The three formats defined for the Control field are used to perform information transfer, basic supervisory control functions, and special or infrequent control functions.

First Bit Transmitted	Control Field							
	1	2	3	4	5	6	7	8
Control Field Bits:								
Information Transfer (I)	0	N(S)			PF	N(R)		
Supervisory Commands/ Response(s)	1	0	S		PF	N(R)		
Unnumbered Commands/ Responses (N)	1	1	M		PF	M		

Where: N(S) = Transmitting station send sequence number.
(Bit 2 = Low Order Bit)

N(R) = Transmitting station receive sequence number.

S = Supervisory function bits

M = Modifier function bits

PF = Poll bit - Primary transmissions.

Final bit - Secondary transmissions.
(1 = Poll/Final)

Details of ADCCP Control are provided by ANSI draft 5 of the
Advanced Data Communications Control Procedure

ASCII

TABLE 3. COMMAND CODE ASSIGNMENTS

Assigned Value	Pneumonic	Description
A	NA	Segment is SOM block
B	NA	Segment is midmessage block
C	NA	Segment is EOM block
D	NA	Single Segment message
E	NA	Segment is EOM/ TX block
F	NA	Single Segment TX block
1	LDRI	Leader Invalid - Source node has rejected the leader for errors
2	TFUL	Tables Full - No connection entry space available. Source Node rejects block
3	BFUL	Buffers Full - Source Node rejects block
4	RFNM	Message Delivered - Request next message
5	HDRI	Header Invalid - Destination Security line violation or Destination Address invalid
6	BFLl	Buffers Full - Destination Node rejects block
7	TIMO	Destination Node has rejected segment due to time out
8	SNSG	Send Next Segment - Source Host can send next segment on this logical channel

5.0 ADPT TESTBED AND EXPERIMENT DESCRIPTIONS

The ADPT testbed program included the development of brassboard hardware and a nodal application software package designed to demonstrate packet switching and voice/data dynamic channel allocation. The current development included hardware development of an Interactive Communications Channel, a Buffer Matrix, and simple Host devices, and software development of packet switch software, including ADCCP line protocol processing and source-destination segment processing, and dynamic channel allocation processing. Dynamic channel allocation and packet switching concepts were successfully demonstrated to the COTR during the acceptance tests.

In addition a network simulator package was designed to load the node and take statistical measurements (thruput, packet delay, trunk utilization, etc). The experiment results will be discussed in section 6. This section describes the testbed set-ups for functional demonstrations and for the experiments.

It should be noted that the ADPT testbed also retains the message switch and circuit switch functions developed in the previous ICMS program and has developed a Pseudo Host software module which facilitates communication between the packet switch (Mode VI) and message switch (Modes I and II) terminals.

The ADPT testbed facility can be configured as two distinct configurations. A three node subnetwork simulator configuration demonstrates the integrated switching of voice and data traffic between unified nodes employing dynamic channel allocation. A test node simulator configuration facilitates experiments designed to measure and analyze node performance in an integrated network environment under various traffic conditions. Figure 5.1 illustrates the physical components comprising the three node subnetwork.

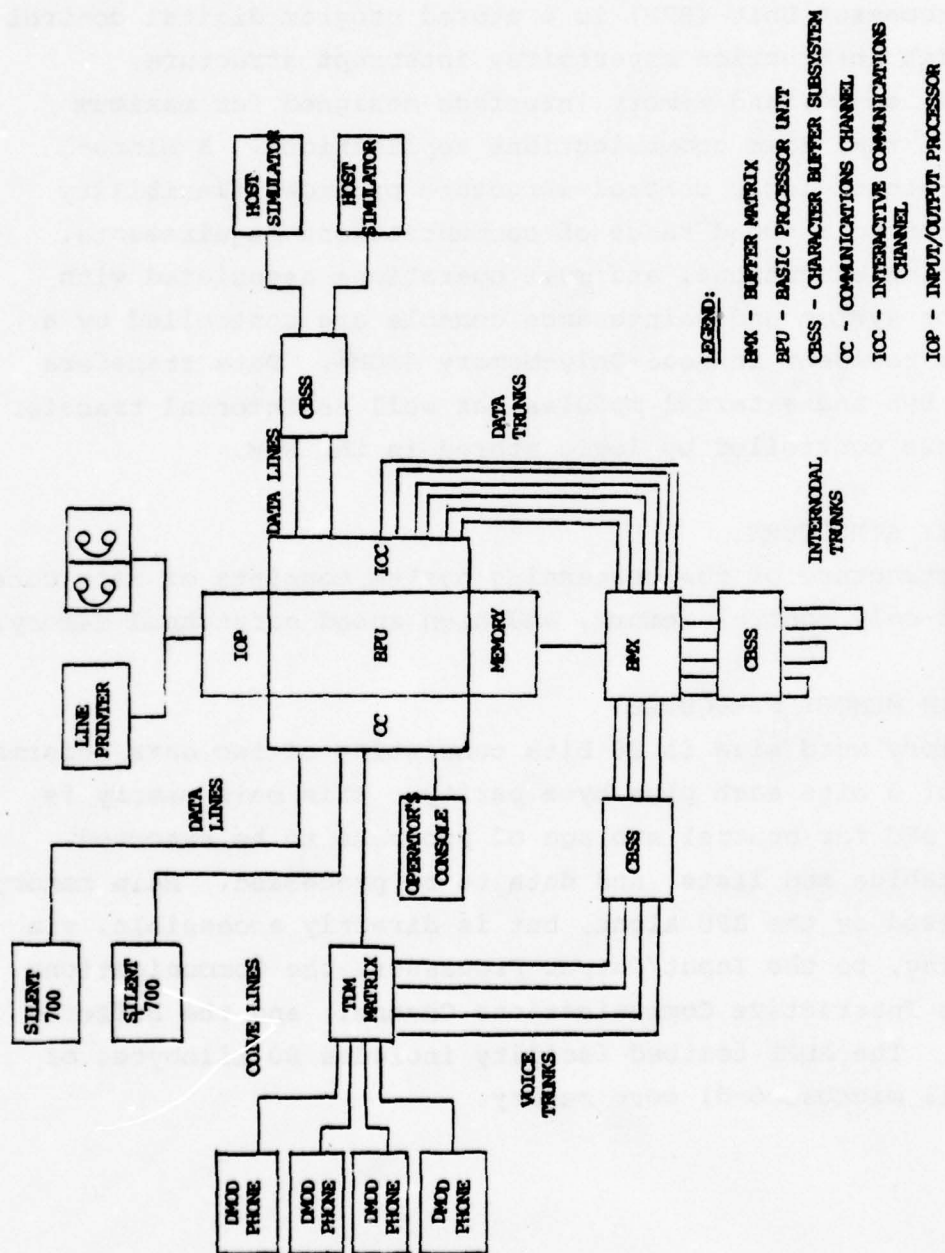


FIGURE 5.1 THREE NODE SUBNETWORK SIMULATOR
(PHYSICAL CONFIGURATION)

5.1 PROCESSING COMPONENTS

The ADPT testbed facility presently utilizes a communications processor, communications channel, an input/output processor, and peripheral devices for both the three node subnetwork simulator and the test node simulator.

5.1.1 COMMUNICATIONS PROCESSOR

The Basic Processor Unit (BPU) is a stored program digital control processor with instruction repertoire, interrupt structure, organization, speed, and memory interface designed for maximum efficiency in real time communications applications. A micro-programmed, stored logic control structure provides flexibility for adaptation to a broad range of communications requirements. All instruction executions, and most operations associated with the interrupt system and maintenance console are controlled by a microprogram resident in Read-Only-Memory (ROM). Data transfers between the BPU and external modules, as well as internal transfer operations are controlled by logic stored in the ROM.

5.1.2 MEMORY STRUCTURE

The memory structure of the processing system consists of main core memory, read-only control memory, and high speed scratchpad memory.

5.1.2.1 MAIN MEMORY STRUCTURE

The main memory word size is 18 bits consisting of two data/information bytes of 8 bits each plus byte parity. This main memory is used by the BPU for central storage of programs to be executed, associated tables and lists, and data to be processed. Main memory is not accessed by the BPU alone, but is directly accessible, via cycle stealing, to the Input/Output Processor, the Communications Channel, the Interactive Communications Channel, and the Buffered Matrix Unit. The ADPT testbed facility includes 80 kilobytes of high speed (1 microsecond) core memory.

5.1.2.2 READ-ONLY MEMORY

The ROM is the control memory of the BPU. This microprogrammed, stored logic control is exerted over all processor functions required by program instruction execution, interrupt facilities, and error control operations.

5.1.2.3 SCRATCHPAD MEMORY

The scratchpad memory logic associated with the BPU provides addressing for 16 general purpose registers (32 bits wide) per state, up to four states. Two states have been implemented for interrupt servicing and other machine functions.

5.1.3 COMMUNICATIONS CHANNEL (CC)

The Communications Channel (CC) provides a real time interface between the BPU and devices that require real time character processing. The CC supports the Mode I and Mode II interface character transfers. The CC can service up to 512 half duplex or 256 full duplex devices at programmable line rates with interrupts being delivered to the BPU on a per character basis. However, the present character buffer subset complement supports only 16 lines.

5.1.3.1 BOOTSTRAP

A paper tape reader is connected to the CC so that a maximum of 4096 bytes of bootstrap program can be loaded into main memory.

5.1.3.2 OPERATOR'S CONSOLE

The operator's console is comprised of a video data terminal and keyboard. It is connected to the CC to provide an operator's interface to initialize and manage operational programs and utility routines within the ADPT testbed facility.

5.1.4 INPUT/OUTPUT PROCESSOR (IOP)

The IOP provides input/output capability for the BPU by receiving generalized input or output instructions from the basic processor and executing them based upon a microprogram resident in the IOP

control memory. The IOP supports data block transfers to and from the peripheral devices. The IOP has direct access to the basic processor's main memory to fetch command words specific to the input/output operation to be performed and to read data from memory that is intended for an output device or to write data into memory that is transferred from an input device. In either case, these memory accesses are effected via memory cycle steals of the basic processor memory. After receiving the start device instruction from the basic processor, the IOP transfers data independent of the basic processor and concurrent with the continued execution of the basic processor's program. Upon completion of the required input/output operation, the IOP signals the basic processor by means of an interrupt that the data transfer is complete. The BPU can now operate on the received data block or account for the transmitted data block.

5.1.4.1 PERIPHERAL DEVICES

Common peripheral devices controlled by the IOP in the testbed configuration include 2 Wang Mod 10 magnetic tape stations and a Data Printer Corp. Model V-132-C medium speed line printer.

5.2 SPECIALIZED SWITCHING COMPONENTS

In addition to common components described in the last section, the ADPT testbed facility includes specialized switching devices in support of the three node subnetwork simulator. The following sections describe the Time Division Multiplex switching matrix, the Interactive Communications Channel, and the Buffer Matrix Unit.

5.2.1 TIME DIVISION MULTIPLEX (TDM) MATRIX

The TDM matrix was developed and tested under the ICMS program. The ADPT testbed facility utilizes the Delta Mod configuration described in the Integrated Circuit/Message Switch Time Division Module Final Report, March 1972, Section III. The TDM matrix rack also contains ancillary hardware for loop supervision and signalling such as register-senders, hook scanner, and digital tone generators.

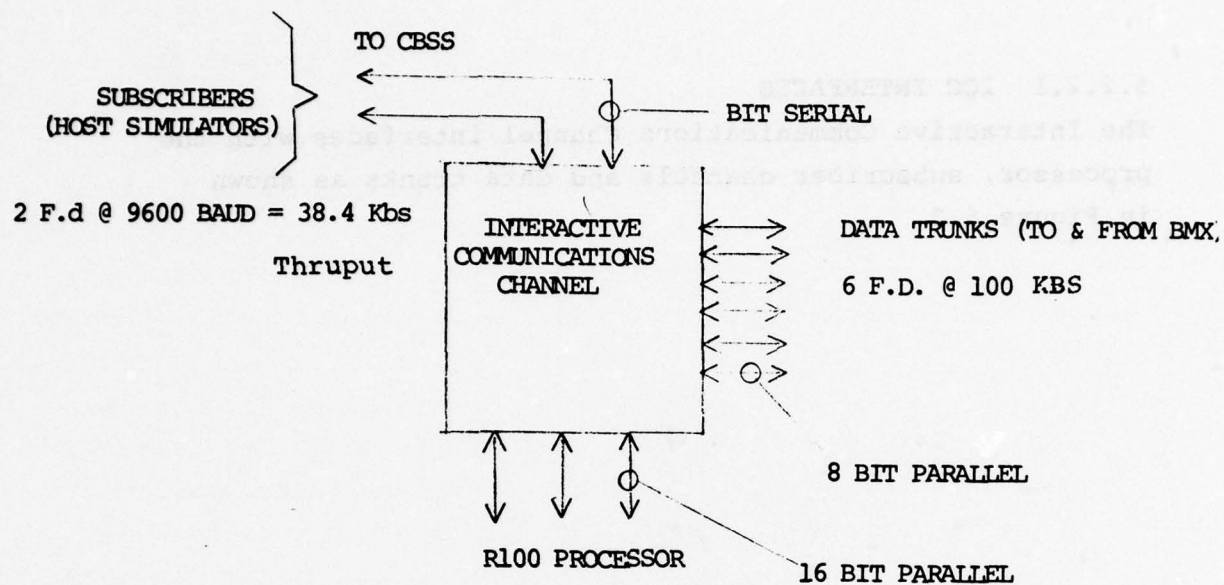
5.2.2 INTERACTIVE COMMUNICATIONS CHANNEL (ICC)

The ICC terminates all packet formatted communications lines within the ADPT testbed facility. Using a combination of hardware and firmware, the ICC performs all bit oriented data processing dictated by the ADCCP and provides the BPU with an interrupt on a per packet basis. Thus, the ICC serves as a front end unit which concentrates data from multiple channels, and also relieves the BPU of a heavy front end processing load. For input channels, the ICC recognizes the start of the packet frame. The ICC collects and stores the incoming data segments (or packets) in preassigned buffers in main core memory. While data is being received, the ICC also strips stuffed zero bits which were added for data transparency, recognizes the end of packet framing sequence, and checks the Cyclic Redundancy Check bits for errors. After completing all operations pertaining to this received packet, the ICC raises an interrupt to the BPU to inform the proper software handler of the reception of a packet and any error conditions which have been encountered.

For output channels, the ICC transmits flag sequences in the absence of data and until the BPU queues data for the channel and activates the channel. The output data contained in buffers in main core memory is obtained and transmitted by the ICC, via direct memory access. While data is being transmitted, the ICC generates any stuffed zero bits needed for data transparency, and computes and transmits the Cyclic Redundancy Check bits. After completing all operations pertaining to this transmitted packet, the ICC raises an interrupt to the BPU to inform the proper software driver of the transmission of packet and any error conditions which have been encountered.

5.2.2.1 ICC INTERFACES

The Interactive Communications Channel interfaces with the processor, subscriber channels and data trunks as shown in Figure 5.2.



F.D. = Full Duplex Channel
 C.B.S.S = Character Buffer Subsystem

FIGURE 5.2 - INTERACTIVE COMMUNICATIONS CHANNEL INTERFACES

The processor communicates with the ICC via the processor memory. Data transfers between the processor and the ICC are 16 bit parallel DMA operations. The processor uses this interface to inform the ICC of buffer areas available for the storage of received data for each channel and the location and length of buffers to be transmitted on each channel. The ICC uses the processor interface to inform the BPU of the complete receipt or transmission of a packet, and error conditions (if they occur). The ICC also provides an interrupt line to the BPU to indicate that a complete packet has been received or transmitted by the ICC.

Character buffers perform bit serial to parallel conversion, so that transfers to and from the ICC are a byte at a time rather than a bit at a time. The Hosts which tie to the character buffers operate at 9600 bits per second. (Full duplex, bit serial)

A special ICC interface terminates six full duplex data channels employing an 8 bit parallel, byte serial interface. These ICC data trunks transmit at a variable data rate as described in Section 2.2.3.1. This interface is between the ICC and the BMX and is implemented so that each channel has the same interface to the ICC as character buffers. These interfaces serve as internal data trunks so that data can be provided to the BMX for multiplexing (with voice) onto the internodal trunks. Each data channel supports one internodal trunk

5.2.3 BUFFER MATRIX (BMX)

The BMX adaptively multiplexes/demultiplexes digitized voice and packet data on all internodal trunks in the ADPT testbed facility. This mixing of voice and data traffic proceeds according to the dynamic channel allocation technique described in Section 5.3.1.1, with the resultant master frame utilized as the basic transmission unit for internodal trunks.

The BMX incorporates a combination of hardwired logic and control

memory to manipulate data within the part of the main memory where all classes of traffic are stored. This memory is accessed by the digitized voice subscribers, the ICC, internodal trunks, and the basic processor. The BPU treats the BMX as just another bank of main core memory. The processor addresses locations in the BMX control memory and reads from or writes to these locations in the same manner as for the shared data memory, with memory contention resolved in hardware by the BMX.

To serve subscriber channels, the processor writes a list of buffer locations and buffer sizes into prescribed locations in the BMX control memory. The BMX logic services each subscriber using a programmed scan order and executes data transfers between subscribers and their designated buffers in memory. These data transfers from or to the memory occur without processor intervention, although the processor can access the memory via cycle stealing to update subscriber control lists. The BMX will process all subscribers in this fashion. Upon reaching the end of the scan, the BMX will loop back to the top of the scan list and continue servicing the subscribers specified.

To serve trunks, the processor writes a frame list of subscribers and their corresponding slot size into prescribed locations in the BMX control memory. This list constitutes a map for the construction of the next master frame to be transmitted on an internodal trunk. In effect, the list contained in the BMX control memory will determine the channel allocation, for subscriber occupancy of a master frame. The BMX will loop back to the top of this control list at the end of a master frame. Two control lists are employed - one for the current master frame, and one for the construction of the map for the next variation of the master frame composition. In the ADPT testbed facility, the control lists are switched only when a digitized voice subscriber is added to or deleted from the master frame. As control lists are switched by the processor, the BMX dynamically changes the channel allocation on the internodal trunk.

5.2.3.1 BMX INTERFACES

The BPU communicates with the BMX via the BMX processor interface. The BPU is capable of addressing all storage locations including voice data memory and BMX control memory. Data transfers between the processor and the BMX are 18 bit parallel operations (2 parity bits and 16 data bits). The BPU uses this interface to establish connections between subscribers and to issue multiplexing and demultiplexing instructions to the BMX for control of the master frame composition on internodal trunks. The BMX also provides an interrupt line to the processor to indicate certain alarm and sync recognition events.

The subscriber BMX interface terminates all Class I subscriber channels. Class I subscribers (CVSD phones) are terminated by four full duplex character buffers at the CBSS which perform serial to parallel conversion of the 32 kilobit per second output of the TDM circuit switch. Six full duplex Class II subscriber channels interconnecting the ICC and BMX employ an 8 bit parallel, byte serial interface. These ICC data trunks transmit asynchronously up to a maximum rate of 100 kilobits per second. The average rate at which data is transferred on these channels varies dynamically and tracks the average rate of input or output on the internodal trunk (which varies dynamically as a function of voice occupancy).

The internodal trunk interface terminates six full duplex character buffers (at the CBSS) which provide parallel to serial conversion. All internodal trunks transmit a bit serial stream at 96 kilobits per second.

5.2.4 CHARACTER BUFFER SUBSYSTEM (CBSS)

The CBSS provides 16 full duplex character buffers and associated clocks for use in the ADPT testbed facility. These character buffers perform parallel to serial and serial to parallel conversion

at various digital rates. Internodal trunks in the three node subnetwork simulator are mechanized by looping the transmit side of one character buffer around to the receive side of another character buffer at the CBSS.

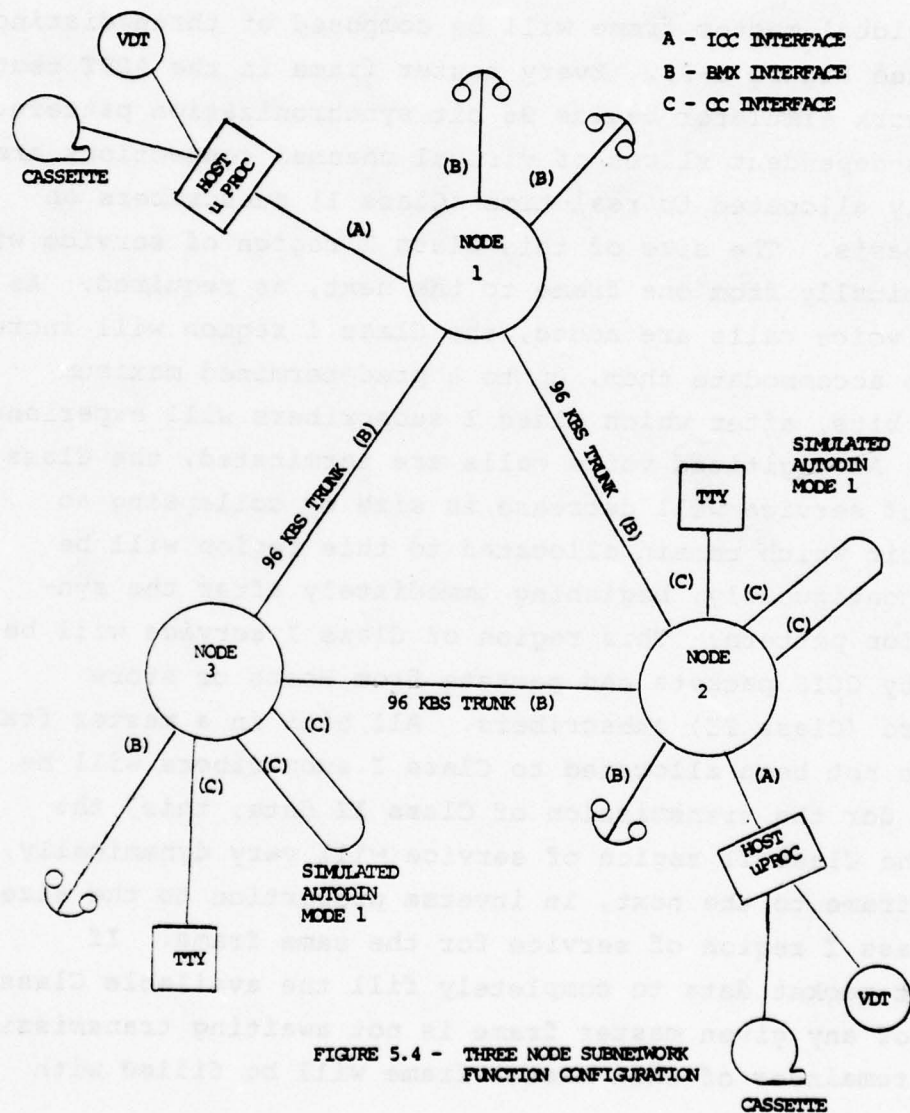
5.3 THREE NODE SUBNETWORK SIMULATOR

The functional capability to operate the ADPT testbed facility in a multinode configuration is provided by the three node subnetwork simulator depicted in Figure 5.1. The equipment complement of the ADPT testbed includes a single communications processor and prototype switching hardware. The three node subnetwork simulator is implemented by using a re-entrant software program to sequentially process three separate sets of tables and queues (one complete set for each node). Switching equipment is shared among the three nodes which comprise the subnetwork and internodal trunks are established by patching physical connections in a back-to-back fashion.

The three node subnetwork simulator was used to validate the functional operation of the specialized switching hardware and applications software of the ADPT testbed facility. This has been accomplished. The three node subnetwork simulator now serves as a vehicle for demonstration of integrated voice/data switching using dynamic channel allocation.

5.3.1 INTEGRATED SWITCHING

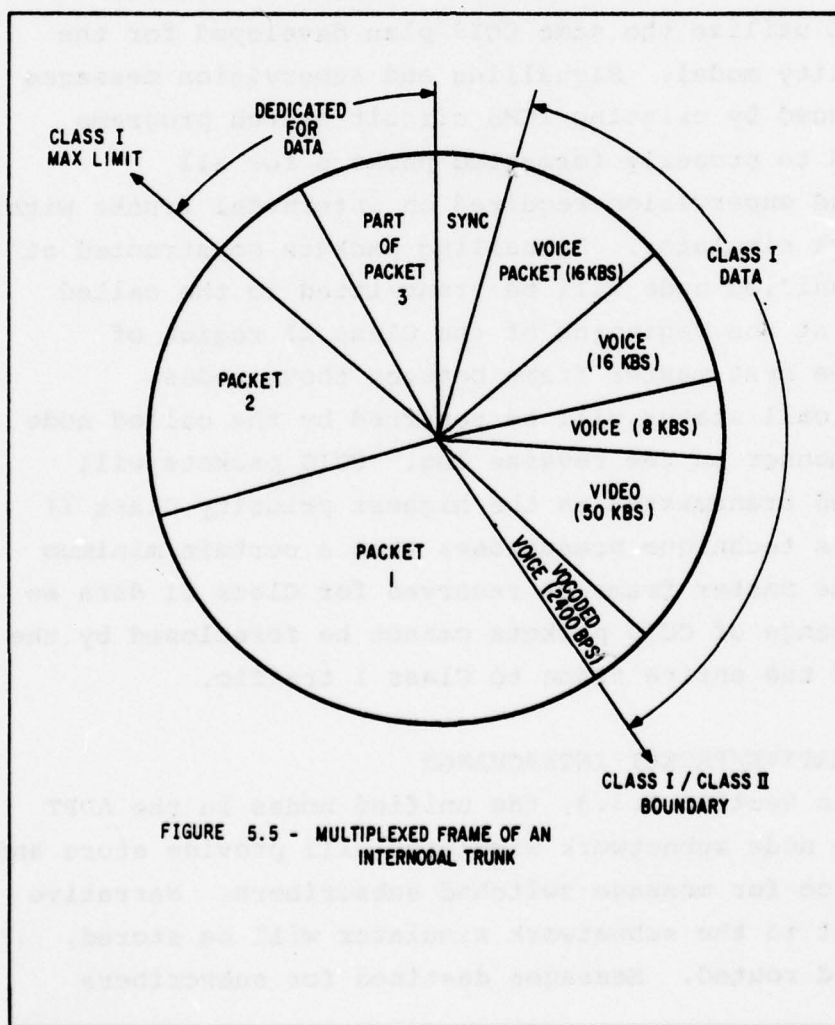
Each node in the three node subnetwork simulator operates as a unified node providing both local access and internodal trunking for a variety of digital subscribers (see Figure 5.4). Within the ADPT testbed facility, local access functions for subscribers to each unified node are provided via circuit switched, store and forward message switched, or packet switched connections, depending on individual subscriber requirements. Both digitized voice and digital data will be trunked between nodes in the subnetwork simulator over a common digital transmission plant employing dynamic channel allocation.



5.3.1.1 DYNAMIC CHANNEL ALLOCATION

All internodal trunks in the three node subnetwork simulator employ dynamic channel allocation for the adaptive multiplexing of digitized voice and digital data onto common transmission facilities. Internodal trunks in the ADPT testbed facility operate at 96 kilobits per second. Digital information (both voice and data) transmitted over these internodal trunks are organized into "Master frames" of 5,760 bits. At the specified trunk transmission rate, $16\frac{2}{3}$ master frames will be transmitted per second.

Each individual master frame will be composed of three distinct regions (see Figure 5-5). Every master frame in the ADPT testbed subnetwork simulator begins 96 bit synchronization pattern. Next, rate-dependent slices of virtual channel connections are dynamically allocated to real-time (Class I) subscribers on a demand basis. The size of this Class I region of service will vary dynamically from one frame to the next, as required. As digitized voice calls are added, the Class I region will increase in size to accommodate them, up to a predetermined maximum number of bits, after which Class I subscribers will experience blocking. As digitized voice calls are terminated, the Class I region of service will decrease in size by collapsing so that all bits which remain allocated to this region will be arranged contiguously, beginning immediately after the synchronization pattern. This region of Class I service will be followed by CCIS packets and packets from Hosts or store and forward (Class II) subscribers. All bits in a master frame which have not been allocated to Class I subscribers will be available for the transmission of Class II data; this, the size of the Class II region of service will vary dynamically, from one frame to the next, in inverse proportion to the size of the Class I region of service for the same frame. If sufficient packet data to completely fill the available Class I region of any given master frame is not awaiting transmission, then the remainder of that master frame will be filled with ADCCP flag sequences.



However, if sufficient packet data is awaiting transmission, complete packets will be transmitted with a partial packet transmitted at the end of the Class II region of service to fill the master frame if necessary. The remainder of any such partial packet will be transmitted at the beginning of the Class II region of service in the next master frame. In the ADPT testbed facility, all packets contained within the Class II region of service will conform to the ADCCP format.

5.3.1.2 COMMON CHANNEL INTEROFFICE SIGNALLING (CCIS)

The three node subnetwork simulator in the ADPT testbed facility will utilize the same CCIS plan developed for the ICMS feasibility model. Signalling and supervision messages will be produced by existing ICMS circuit switch programs and converted to properly formatted packets for all signalling and supervision required on internodal trunks within the subnetwork simulator. Signalling packets constructed at the calling unified node will be transmitted to the called unified node at the beginning of the Class II region of service of the next master frame between these nodes. Progress and call status will be returned by the called node in the same manner on the reverse leg. CCIS packets will be handled and transmitted as the highest priority Class II traffic. This technique presupposes that a certain minimum portion of the master frame be reserved for Class II data so that the exchange of CCIS packets cannot be foreclosed by the allocation of the entire frame to Class I traffic.

5.3.1.3 NARRATIVE/PACKET INTERCHANGE

As outlined in Section 5.3.3, the unified nodes in the ADPT testbed three node subnetwork simulator will provide store and forward service for message switched subscribers. Narrative messages input to the subnetwork simulator will be stored, validated, and routed. Messages destined for subscribers

serviced by another unified node in the three node subnetwork simulator will be broken up into segments which will not exceed three line blocks (240

characters) in length. Each of these segments will be converted to a properly formatted packet by the addition of an appropriate packet header and ADCCP fields. These packets will then contend with other packet data for transmission over an internodal trunk during the Class II region of service in a subsequent master frame. At the destination unified node, disposition of the narrative message will be made based on destination terminal type. If the message addressee is a store and forward subscriber, the destination unified node will strip the ADCCP fields and packet headers, store the entire narrative message, and deliver the message to the destination terminal. If the message addressee is a packet subscriber, the destination unified node will deliver each received packet intact to the destination terminal.

5.3.2 CIRCUIT SWITCH

Unified nodes in the ADPT testbed three node subnetwork simulator will utilize the Time Division Multiplex (TDM) portion of the circuit switching hardware and software developed under the ICMS project. The TDM circuit switch will provide local access loops and ancilliary equipment such as hook scanner, tone generators, and digit registers for digitized voice subscribers. Both local and remote connections will be established via the TDM matrix under the control of the R100 communications processor.

5.3.2.1 LOCAL CALLS

Local calls between digitized voice subscribers connected to the same unified node in the subnetwork simulator will be handled and completed via the TDM matrix in exactly the same manner as demonstrated in the ICMS feasibility model.

5.3.2.2 REMOTE CALLS

Calls for which either the calling or called digitized voice subscriber is connected to a distant unified node in the subnetwork simulator will involve two local connections via the TDM matrix and an internodal trunk connection. The calling digitized voice subscriber will be connected to a port on the adaptive multiplex

device (the BMX) via the TDM matrix, just as if this port were a local digitized voice subscriber. At the other unified node, the called digitized voice subscriber will be connected to a port on the BMX via the TDM matrix, again, just as if this port were a local digitized voice subscriber. The connection between the calling and called unified nodes will be established by the allocation of a rate-dependent number of bits in the Class I region of each master frame transmitted on the internodal trunk linking the nodes. The position of the bits allocated to this call in the Class I region of service may vary from one master frame to the next as other calls terminate. Nonetheless, the same number of bits will remain allocated for the duration of the call. Taken together, these connections will form a complete digital transmission path between the calling and called digitized voice subscribers.

5.3.2.3. TANDEM CALLS

Calls for which both the calling and called digitized voice subscribers are connected to distant unified nodes in the subnetwork simulator will involve two local connections via the TDM matrix and two internodal trunk connections. As in the remote case, the calling and the called digitized voice subscribers will be connected to respective ports on the BMX via the TDM matrix as described in Section 5.3.2.2. The connection between the calling and the tandem unified nodes will be established by the allocation of a rate-dependent number of bits in the Class I region of service of each master frame transmitted on the internodal trunk linking these nodes. The connection between the tandem and the called unified nodes will be established by the similar allocation of bits in the Class I region of service of each master frame transmitted on the internodal trunk linking these nodes. The position of the bits allocated to this call in the Class I region of service may vary independently on both internodal trunks from one frame to the next as before. Nonetheless, the same number of bits will remain allocated on both internodal trunks linking the calling, tandem, and called nodes. Taken together, these connections will form a

complete digital transmission path between the calling and called digitized voice subscribers. Note that the TDM matrix at the tandem node will not be involved in the connection of a tandem call.

5.3.3 MESSAGE SWITCH

Store and forward message switch subscribers will be connected to the ADPT testbed three node subnetwork simulator by means of dedicated circuits. The two ASR Silent 700 terminals included in the testbed facility equipment complement will be used to demonstrate both store and forward message switching and the narrative/packet interchange described in Section 5.3.1.3.

5.3.3.1 PROTOCOLS AND FORMATS

Messages originated by store and forward subscribers in the three node subnetwork simulator will conform to the format requirements set forth in JANAP 128 (D). Validation of messages for proper format will be accomplished at the unified node to which the originating subscriber is connected. As a minimum, each message must contain the elements of the abbreviated plain-dress option permitted under JANAP 128 (D). Because of the characteristics of the available Silent 700 equipment, these devices will operate asynchronously as AUTODIN Mode II terminals, in exactly the same manner as demonstrated in the ICMS feasibility model by ASR 28 Teletypes.

5.3.3.2 PRIORITY HANDLING

Messages received at a unified node from store and forward subscribers in the three node subnetwork simulator will be stored, validated, and routed on a first in, first out basis without regard to precedence. However, messages awaiting output to store and forward subscribers by a unified node will be transmitted based upon five precedences: Emergency (Y), Flash (Z), Immediate (O), Priority (P), and Routine (R).

5.3.3.3 SECURITY

The message switching portion of each unified node within the three node subnetwork simulator will examine messages destined for store and forward subscribers and insure that a specific terminal is authorized to receive the appropriate level of classified traffic before a classified message will be delivered to that terminal. Messages bearing any of five security classifications will be rejected for exceeding a terminal's security authorization: Top Secret (T), Secret (S), Confidential (C), Encrypted for transmission only (E), and Unclassified (U).

5.3.3.4 ROUTING

Routing of store and forward messages by the message switching portion of a unified node within the three node subnetwork simulator will be accomplished on the basis of a table lookup of the seven letter routing indicator contained in Format line 2 of the message. A maximum of two routing indicators will be processed per message.

5.3.4 PACKET SWITCH

Packet switch subscribers will be connected to the ADPT testbed three node subnetwork simulator by means of dedicated circuits. The two Host simulator terminals included in the testbed facility equipment complement will be used to demonstrate packet switching.

5.3.4.1 PROTOCOLS AND FORMATS

Messages originated by packet subscribers in the three node subnetwork simulator will be transmitted as segments which conform to the format requirements set forth in the American National Standards Institute's proposed "Advanced Data Communications Control Procedures" (ADCCP), Fifth Draft edition. Segments transmitted over the communications link between a Host simulator and the packet switch portion of the connected unified node will have flag sequence generation/detection, bit stuffing/destuffing, and CRC accumulation/checking performed by a combination of hardware and firmware at each end of the link.

This communication link will operate synchronously at 9.6K bits per second with flag sequences being continuously transmitted in the absence of data. In the ADPT testbed, segments will be restricted to a maximum length which coincides with the maximum packet size of 240 characters. Validation of segments for proper format and associated protocol will be accomplished at the unified node to which the originating subscriber is connected.

Segments which will be transmitted over internodal trunks will first be converted to packet format at the originating unified node. Packets derived in this fashion will be transmitted during the Class II region of service in a subsequent master frame between the nodes involved. At the destination unified node, these packets will be converted back into segments and delivered to the destination packet switch subscriber terminal.

5.3.4.2 PRIORITY HANDLING

Segments and packets will be processed in the order of receipt on the access side of the packet switch portion of unified nodes in the ADPT three node subnetwork simulator. However, packets awaiting output on dynamically allocated internodal trunks will contend for transmission in the Class II region of service in a master frame based upon category. CCIS packets and protocol response/control packets (Category A) will be handled first, followed by interactive data and emergency or flash precedence narrative data packets (Category B). Next, all other precedence narrative data packets (Category C) will be transmitted, followed by bulk data (Category D) traffic.

5.3.4.3 SECURITY

The packet switching portion of each unified node within the three node subnetwork simulator will examine packets and segments destined for packet switch subscribers and insure that a specific terminal is authorized to receive the appropriate level of classified traffic before a classified segment will be delivered to that

terminal. Packets and segments bearing any of the five security classifications enumerated in Section 5.3.3.3 will be rejected for exceeding a terminal's security authorization.

5.3.4.4 ROUTING

Routing of packets by the packet switching portion of a unified node within the three node subnetwork simulator will be accomplished on the basis of the address field contained in the packet header and a table lookup. Only a single addressee will be permitted in each packet, but alternate routing will be invoked by failure or congestion of an internodal trunk within the subnetwork simulator.

5.4 TRAFFIC SOURCES AND SINKS

The equipment complement of the ADPT testbed facility includes a variety of traffic sources and sinks for the three node subnetwork simulator. These devices are employed as circuit switch, message switch, and/or packet switch subscribers to demonstrate the operation of unified nodes and dynamic channel allocation within the three node subnetwork simulator.

5.4.1 CVSD PHONES

Four GFE Continuously Variable Slope Delta (CVSD) modulated telephones will be used as digitized voice circuit switch subscribers in the three node subnetwork simulator. The phones are connected in pairs to two of the subnetwork nodes via the TDM matrix. The CVSD phones operate at 32 kilobits per second. Local, remote, and tandem digitized voice calls are originated and terminated by these phones (as described in Section 5.3.2).

5.4.2 SILENT 700 TERMINALS

Two Silent 700 terminals will be used as store and forward message switch subscribers in the three node subnetwork simulator. Each Silent 700 terminal will be connected to a different subnetwork node by means of a dedicated circuit. These terminals will operate

asynchronously in the full duplex mode at 150 baud using an eight level ASCII character set. Narrative messages will be originated by and delivered to the Silent 700 terminals as described in Section 2.3.3.

5.4.3 HOST SIMULATORS

Two Host Simulator terminals will be used as packet switch subscribers in the three node subnetwork simulator. Each Host terminal will be connected to a different subnetwork node by means of a 9600 bit per second synchronous ADCCP link. Segments will be originated by and delivered to the Host Simulator terminals as described in Section 2.3.4.

5.5 TEST NODE SIMULATOR

The functional capability to use the ADPT testbed facility to measure and analyze nodal performance parameters will be provided by the test node simulator. The test node simulator will be implemented using only the common components described in Section 5.1 including the BPU, memory, communications channel with operator's console, and IOP with two magnetic tape stations and medium speed line printer. The functions of the specialized switching components will be simulated in order to free the test node simulator from the constraints inherent in the three node subnetwork simulator. Specifically, to obtain the traffic intensities planned for the test node simulator would require hundreds of phones and packet terminal devices. By simulating the ICC and the BMX, the test node simulator will be able to sustain sufficient traffic loads to stress the nodal applications program.

5.5.1 TEST NODE SIMULATOR SOFTWARE PACKAGE

While the test node simulator equipment configuration will be relatively straightforward, the software package which will operate on this equipment will be rather sophisticated. This software package will be logically divided into microscopic and macroscopic parts. The microscopic portion of the test node

simulator will be derived from the validation applications software of the three node subnetwork simulator, while the macroscopic portion will functionally represent the surrounding network.

5.5.1.1 MICROSCOPIC SIMULATION

The reentrant program which will sequentially process three sets of tables and queues in the three node subnetwork simulator and this program's relationship to the ICC and BMX is shown in Figure 5.6. After the functional operation of the three node subnetwork simulator has been verified, this same applications software program will be set up to process one set of tables and queues and included in the test node simulator software package.

5.5.1.2 MACROSCOPIC SIMULATION

While the microscopic portion of the test will consist of operational software, the macroscopic portion will provide the functional equivalent of a connected network environment. The macroscopic portion of the test node simulator will be used to drive the microscopic nodal software and to facilitate the measurement of nodal performance parameters. The relationship of the macroscopic simulation modules to the microscopic test node applications software is shown in Figure 5.7. The following sections will describe the operation of the test node simulator including the off-line generation of the source traffic tape, the on-line processing of voice and data traffic, and the collection of parametric data on the test node's performance.

5.5.2 TRAFFIC GENERATION

To supply sufficiently large traffic loads to adequately stress the test node simulator, compressed voice and data transactions will be introduced by means of a prerecorded source traffic tape. A typical traffic tape will contain intermixed voice and data transactions in order of the desired time of entry to the test node simulator. The source traffic tapes used will be produced off-line by a discrete-event simulation program (written in GPSS)

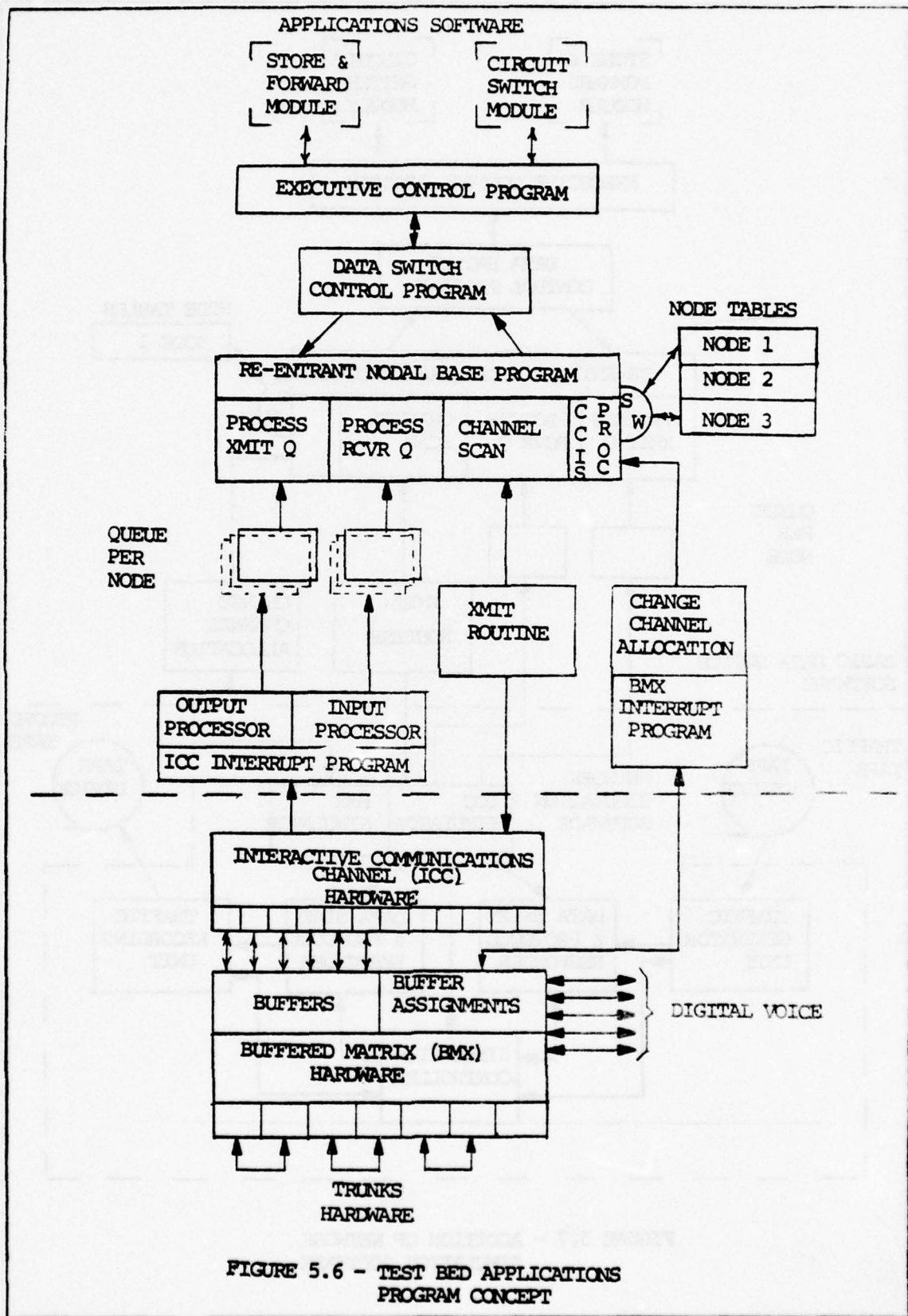
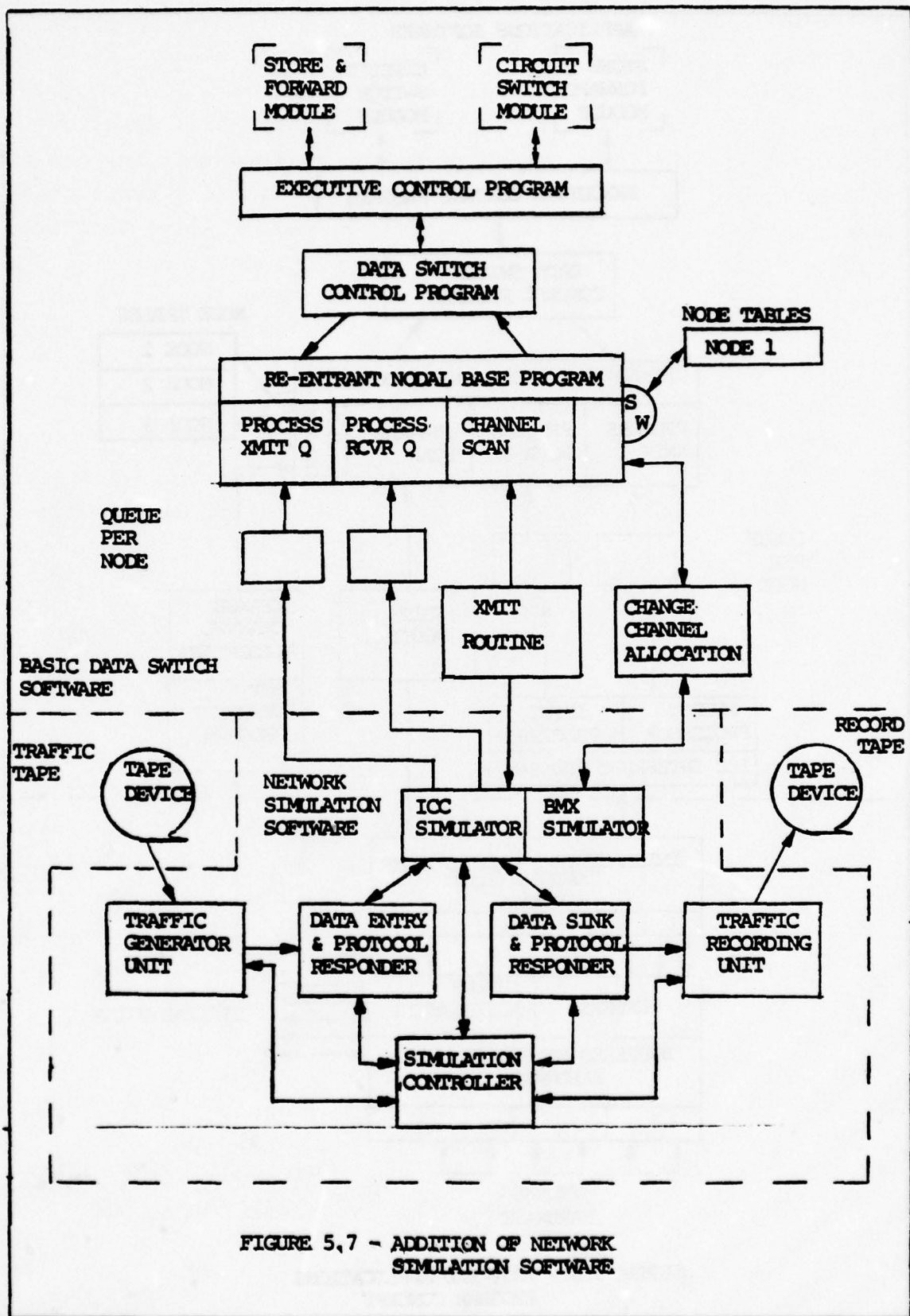


FIGURE 5.6 - TEST BED APPLICATIONS PROGRAM CONCEPT



based on user specified transaction interarrival times, length/bandwidth distributions, precedence mixes, and call holding times. The resulting source traffic tapes will contain three types of transaction descriptors: voice initiate, voice terminate, and data. Each descriptor on the traffic tape will contain only transaction identification and control information; no text will be included in these descriptors. Transaction length (for data) or bandwidth (for voice) will be specified by the appropriate transaction descriptor. The macroscopic portion of the test node simulator will use the length of data transactions and the bandwidth of voice transactions to simulate the corresponding traffic load.

In the case of voice transactions, separate transaction descriptors will indicate the beginning and the end of a voice call. Voice transactions will be used to modulate the internodal trunk bandwidth available for data in the test node simulator. In the case of data transactions, the transaction descriptor includes a skeleton segment/packet header and information specifying the number of segments/packets and the size of the last segment/packet in the transaction. Data transaction descriptors will be used by the macroscopic network simulation software to generate individual segments or packets which will drive the test node access lines and trunks according to data flow control commands and requests issued by the microscopic test node software. For example, the first segment or packet of a data transaction will be sent to the test node unsolicitedly, while subsequent segments or packets in this transaction will be called for by the microscopic test node software, as required.

5.5.3 NETWORK SIMULATION SOFTWARE

As illustrated in Figure 5.8, the test node will be surrounded by the network simulation software which will provide the functional equivalent of a connected network environment. The validated applications modules derived from the three node

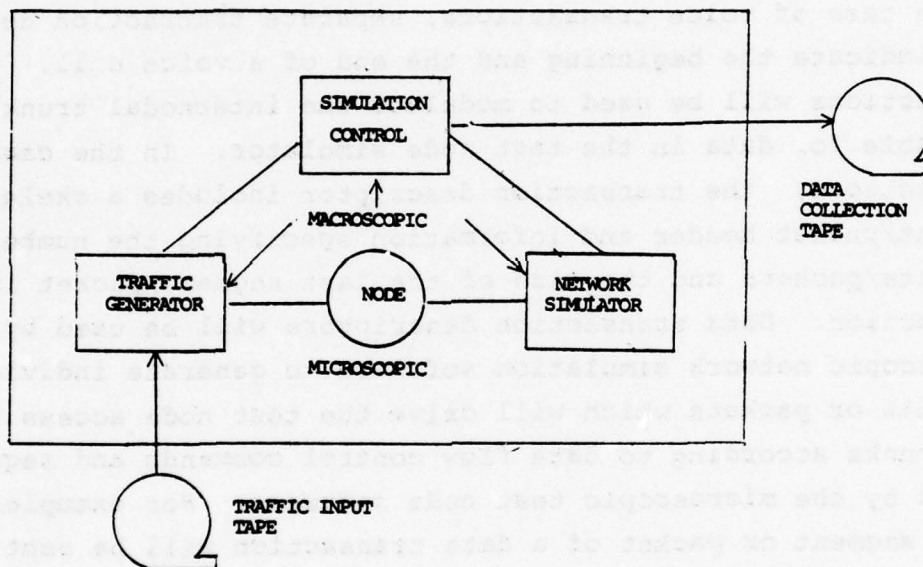


FIGURE 5.8 TEST NODE SIMULATOR SOFTWARE

subnetwork simulator will represent the detailed operational functions of the microscopic portion, the test node, while the network simulation modules will comprise a macroscopic simulation of the surrounding network. The network simulation modules will be discussed in the following sections.

5.5.3.1 SIMULATOR CONTROL UNIT (SCU)

SCU will coordinate the execution of the other network simulation modules, and manage buffer resources allocated to the network simulation software for storage of transactions in progress. In addition, SCU will manipulate the elapsed time counter and the software clock. This will involve passing control to other simulation modules and to the nodal applications program. The nodal applications program will run for a programmable period of time, such as a millisecond, and the network simulation program will run for as long as required to gather test data and perform required I/O operations. The "run time" of the test node will be frozen while simulation programs are active, so that run times will not distort nodal delay measurements. Figure 5.9 depicts a typical timing sequence illustrating the internal progression of processing events within the processor. Hardware priority interrupts will serve to transfer control from the application program to the simulation program at the expiration of the elapsed time counter.

5.5.3.2 TRAFFIC GENERATOR UNIT (TGU)

TGU is the network simulation software module which will read traffic blocks from the source traffic tape. These traffic blocks will contain descriptors which represent voice and data transactions. TGU will interrupt each traffic descriptor, initiate both voice and data transaction at predetermined times, and terminate voice transactions as required.

TGU will initiate voice transactions by subtracting the bandwidth specified in the traffic descriptor from the total bandwidth available on the appropriate internodal trunk(s). If sufficient

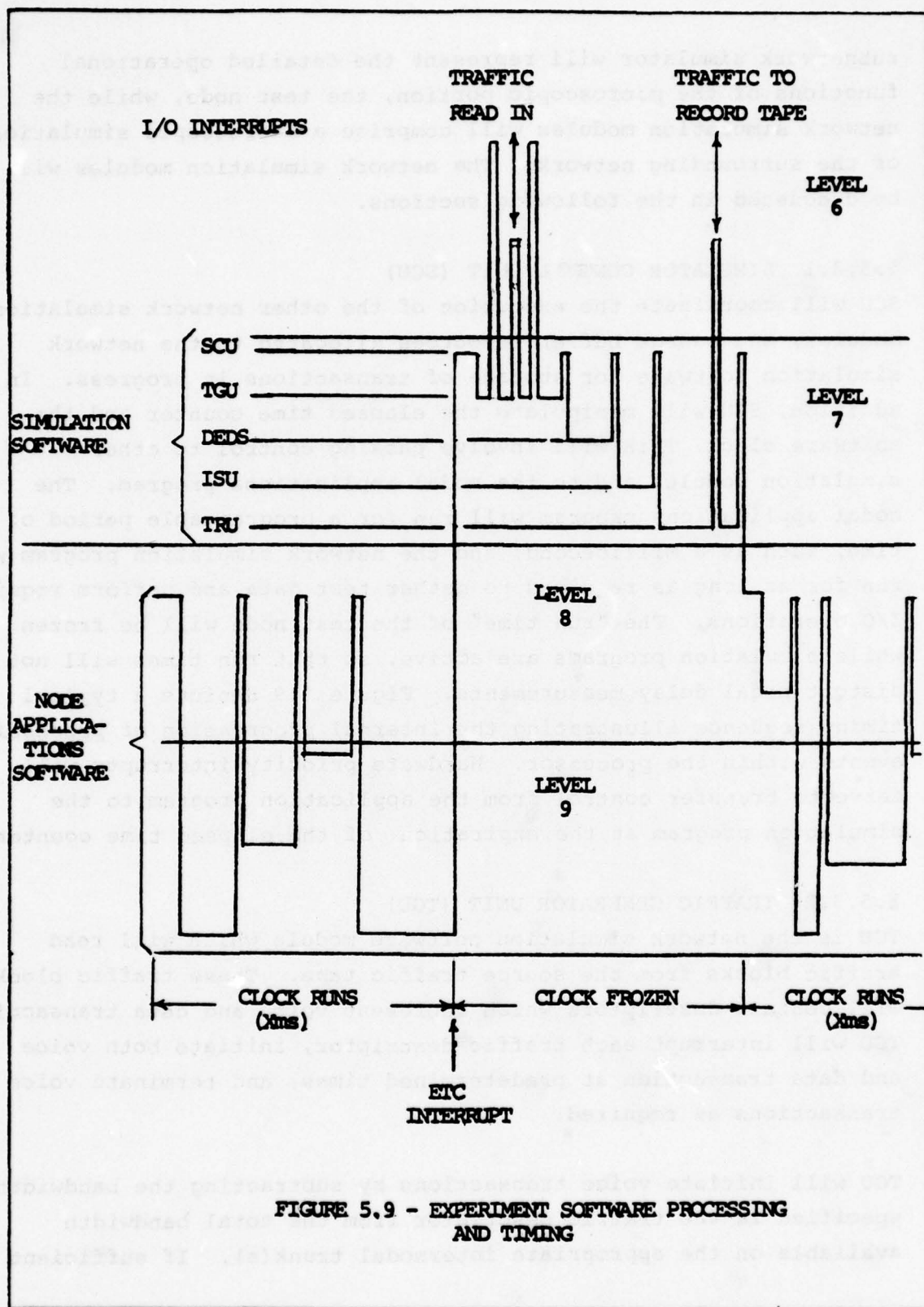


FIGURE 5.9 - EXPERIMENT SOFTWARE PROCESSING AND TIMING

bandwidth to complete the call is not available on the specified trunk(s), the call will be blocked and cleared. TGU will terminate voice transactions by adding the call bandwidth to the total bandwidth available on the appropriate internodal trunk(s).

At the appropriate time, TGU will generate a properly formatted "Start of Message" packet or segment for all data transactions according to the information specified in the traffic descriptor. The first packet/segment will then be queued to the selected access line or trunk for entry into the test node.

5.5.3.3 DATA ENTRY/DATA SINK (DEDS)

DEDS is the module which will receive and interpret responses sent by the test node and generate properly formatted "Middle of Message" and "End of Message" packets or segments based upon these responses and information contained in the transaction descriptor. Thus, DEDS will continue, and eventually will terminate, all data transactions initiated by TGU.

Unlike TGU, which will be driven by discrete traffic events recorded on the source traffic tape, DEDS will react to actual traffic sent by the test node. Depending upon the packet or segment received, DEDS will generate a command packet or the next sequential packet or segment of the data transaction, which will then be queued to the proper access line or trunk for entry into the test node. Upon recognition of the last packet of a message, DEDS will prepare a transaction termination record for output to the traffic record tape.

5.5.3.4 ICC SIMULATOR UNIT (ISU)

This module replaces the hardware Interactive Communications Channel (ICC) device whose function was to introduce and remove packets and/or segments to and from the test node. By simulating the ICC, physical limitation on traffic loading will be removed. ISU will provide the interface between the test node applications program and the network simulation software. Just as DEDS carried

on the end-to-end protocol by responding to packets sent from the test node, ISU will provide the ADCCP link protocol for all test node access lines and internodal trunks.

ISU will function in precisely the same manner as the hardware device it will replace. Upon the complete insertion or extraction of a data packet to/from the test node, a software controlled interrupt will be generated. The nodal applications program will service this interrupt as if the physical ICC had generated it.

ISU will also control the delays associated with the simulated network and measure the processing delays incurred by packets sent to the test node. Packet delay records will be prepared for output to the traffic record tape as packets exiting the test node are encountered.

5.5.3.5 TRAFFIC RECORD UNIT (TRU)

TRU is the vehicle by which measurements of critical nodal parameters will be collected from the various other network simulation modules, formatted, and written to the traffic record tape. TRU will handle both transaction termination records and periodic performance statistics. Traffic termination records will be produced whenever a voice or data transaction is completed (whether normally or abnormally). Performance statistics will be compiled from various tables and counters throughout the network simulation software and written to the traffic record tape in prescribed format at periodic intervals which will be specified by the test operator at program run time. The traffic record statistics will be reduced off-line to produce desired report data for evaluation and analysis.

5.6 EXPERIMENT VARIABLES

The measurement of nodal thruput, delays, and similar nodal characteristics will include a range of situations produced by varying certain critical nodal and traffic parameters from one

simulation run to the next. A comparison of the data generated over a series of runs will yield information which can be interpreted as an indication of the sensitivity of the test node to the parameters involved. The parameters which will be varied for these experiments include: traffic load, traffic composition, line and trunk rates, number of lines and trunks, voice grade of service, routing doctrine, and throttling rules. A further description of these variables is contained in the ADPT Experiment Plan (CDRL Item A006).

6.0 ANALYSIS AND EVALUATION OF EXPERIMENTAL RESULTS

Following the demonstration of the ADPT testbed facility to validate the functional performance of the applicable hardware and software, the microscopic portion of the node simulator was integrated with a macroscopic simulation representing the surrounding network environment. Several short experiments were conducted to gain experience with the operation of the test simulator and to facilitate the adjustment of nodal parameters for the planned experiments. Five full length experiments were performed and the results are included in this report as Appendix III.

Detailed computer printouts pertaining to trunk utilization, completed transaction histories, queue behavior, and thruput were obtained for each of the five experiments; this data is not included in appendix III because this raw data is voluminous. The remaining paragraphs of this section are devoted to the presentati analysis, and evaluation of the results generated by the five experiments which were performed.

6.1 EXPERIMENT DESCRIPTION

The microscopic nodal program component of the ADPT node simulator is almost identical to the nodal operation demonstrated to witnesses from the Rome Air Development Center (RADC). There are, however, minor differences beyond the replacement of specialized switching hardware by simulation software which are worthy of mention. Foremost among these differences is a modification to the data switch software control structure. In the demonstration software package, the data switch control program was organized to process transmit queues ahead of the receive queue. Under the essentially no-load conditions created during the operation of the demonstration version of the ADPT testbed facility, this circumstance posed no problems. However, when subjected to significant traffic loads, this algorithm gave rise to random failures related to the ADCCP link protocol numbering scheme and the mechanism of packet/segment acknowledgment. Using the capabilities of the ADPT node simulator, this sophisticated problem was diagnosed, analyzed, corrected, and tested. The solution to this problem

was to modify the control algorithm to process the receive queue ahead of the output queues.

6.1.1 NODE SIMULATOR CONFIGURATION

The macroscopic and microscopic portions of the ADPT testbed node simulator were assembled together with both fixed and variable parameters. The resulting node simulator package represents the nodal configuration used to conduct the five experiments. The node simulator has 15 access lines, of which only 12 were used, plus a single trunk. The access lines were set to operate at 192 kilobits per second (full duplex) while the trunk was set to operate at 1.544 megabits per second (full duplex), the T1 carrier rate. The trunk was limited by a programmable parameter to a maximum of 1.408 megabits per second of voice traffic which produced a blocking probability of 2.1% for voice subscribers over the run period of one hour. (The theoretical blocking probability is 1%.) A constant 100 millisecond delay was added to data packet responses transmitted on the input internodal trunk, to simulate network turnaround delay. Figure 6.1 depicts the node simulator configuration used. Operation of the ADPT node simulator is discussed in the next paragraph.

6.1.2 NODE SIMULATOR OPERATION

The operational performance record of the ADPT node simulator for each conducted test experiment is summarized in Table 6.1. Each of the five experimental runs shown in the table is numbered according to input traffic volume and time scale factor. Three different input traffic tapes were employed in the course of the five experiments, but each represented the same voice traffic load of approximately 72 Erlangs, while the intensity of data traffic varied. Run #1 involved roughly 12,500 data transactions, Runs #2A, 2B, and 2C used approximately 25,000 data transactions, and Run #3 processed almost 50,000 data transactions. The time scale factor used during each of the five experiments varied from 1 to 4, simulating machines with the processing speed of the basic R100 to computers four times faster. The wall clock time taken to simulate one hour of node operation is given in the table

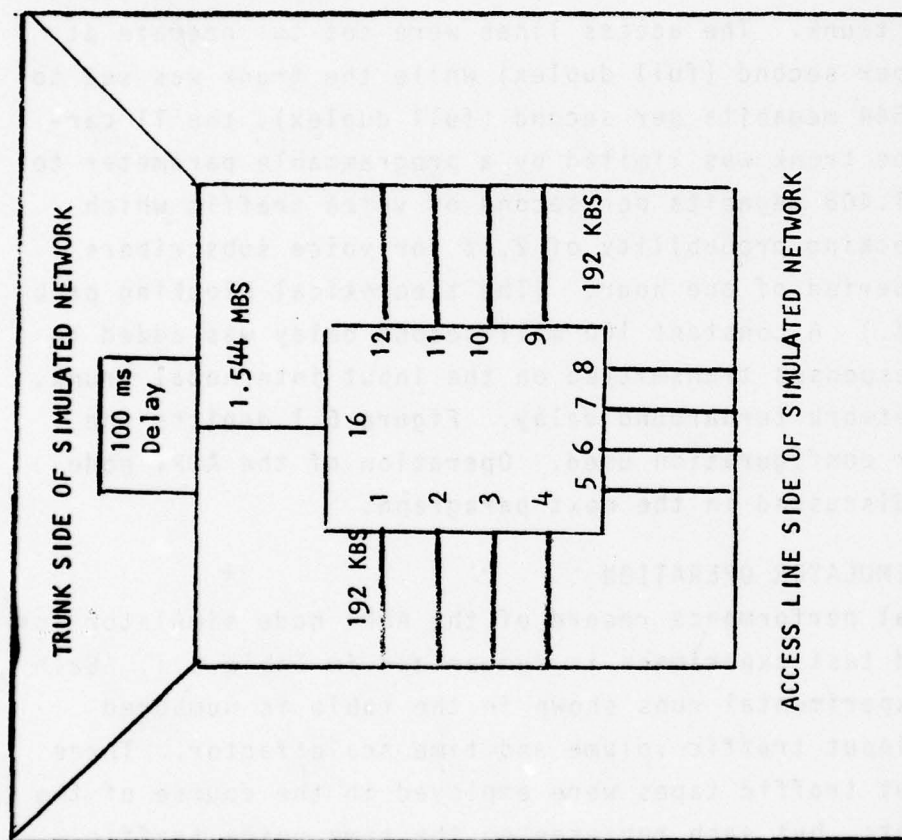


FIGURE 6.1 NODE SIMULATOR CONFIGURATION

RUN NUMBER	DATA TRAFFIC VOLUME	TIME SCALE FACTOR	RUN TIME	DATA REDUCTION TIME
1	12,500	1	275 min.	97 min.
2A	25,000	1	330 min.	153 min.
2B	25,000	2	360 min.	170 min.
2C	25,000	4	480 min.	180 min.
3	50,000	4	600 min.	315 min.

NOTE - Voice traffic volume = 72 ERLANGS (16KBS per channel)

TABLE 6.1 NODE SIMULATOR PERFORMANCE RECORD

for each run. The ratio of wall clock time to nodal simulation time ranged over values from approximately 4.5 to 1 to 10 to 1. Variation in the time required to run data reduction programs associated with each experiment are primarily due to the quantity of output tape processed and the number of pages printed. These parameters are related to the number of data transactions involved in each experiment. Omission of the detailed completed transaction histories and second by second accounts of trunk utilization, queue behavior, and thruput can substantially reduce the data reduction time for a given experimental run. Decreased resolution of the simulation clock and additional buffering (batching) for tape I/O operations can be employed to reduce the wall clock time necessary to run a given experiment and to improve the real to nodal time ratio.

6.2 EXPERIMENTAL RESULTS

The quantitative results of the five experimental runs are summarized in Table 6.2 Each run is numbered according to the volume of data transactions and the time scale factor, as before. For each run, the information presented includes the total number of packets, the mean and maximum tandem packet delay in ms for each of the four data priorities: control (C), and priority 1-3 (P1, P2, P3), the mean and maximum nodal thruput in packets per second or segments in and out, the mean and maximum queue size in number of packets for the trunk transmit queue (X) and the four associated backlog queues (control, and priority 1-3), the mean and maximum trunk utilization in percent, for input, output, and voice, the mean and maximum buffer utilization (in percent), and the the average processor utilization (in percent). The information contained in this table was obtained from the basic statistics report generated by the data reduction program. Probability density functions and cumulative probability distribution functions are available from the

advanced statistics report optionally generated by the data reduction program. These reports are included in Appendix III. In addition, information concerning completed data transactions, as well as a second by second account of trunk utilization, voice calls and data transactions in progress, queue behavior and thruput were generated. Figures

RUN NUMBER	TOTAL NUMBER OF PACKETS	TANDEM PACKET DELAY		THROUGHPUT MEAN MAX	QUEUE SIZE		TRUNK UTILIZATION (%)		BUFFER UTILIZATION MEAN MAX	PROCESSOR UTILIZATION
		MEAN	MAX		MEAN	MAX	MEAN	MAX		
1	356,736	C	27.8 153	INPUT 80.9 139	X	1.02 7	INPUT 71.9 98.4		46.5 83	71.4%
		P1	32.0 195		C	.26 5	OUTPUT 71.7 97.4			
		P2	32.4 230	OUTPUT 99.1 172	P1	.16 4	VOICE 67.3 91.2			
		P3	32.8 233		P2	.06 3				
2A	495,744	C	50.1 —	INPUT 112.7 149	X	3.15 7	INPUT 73.6 98.7		101.6 138	86.0%
		P1	58.0 —		C	.70 7	OUTPUT 73.4 97.9			
		P2	60.2 —	OUTPUT 133.7 177	P1	.44 7	VOICE 67.3 91.2			
		P3	69.8 —		P2	.18 4				
2B	741,760	C	16.3 102	INPUT 168.4 250	X	1.95 7	INPUT 77.2 99.8		68.8 117	71.9%
		P1	20.7 151		C	.32 4	OUTPUT 76.5 100.0			
		P2	21.9 296	OUTPUT 206.0 301	P1	.19 5	VOICE 67.3 91.2			
		P3	22.8 1492		P2	.07 4				
2C	742,272	C	5.9 97	INPUT 168.5 364	X	1.32 7	INPUT 77.2 100.0		63.4 118	28.0%
		P1	10.2 137		C	.10 3	OUTPUT 76.5 100.0			
		P2	10.9 827	OUTPUT 26.2 458	P1	.06 4	VOICE 67.3 91.2			
		P3	10.6 1859		P2	.025 11				
3	1,414,270	C	9.5 83	INPUT 322.2 452	X	3.40 7	INPUT 86.1 99.9		103.3 185	62.7%
		P1	13.8 111		C	.30 3	OUTPUT 84.9 100.0			
		P2	14.2 197	OUTPUT 392.9 546	P1	.16 5	VOICE 67.3 91.2			
		P3	14.5 529		P2	.07 6				

TABLE 6.2 EXPERIMENTAL RESULTS

6.2 and 6.3 are good examples of the type of information that can be obtained from these detailed data reduction reports. Figure 6.2 is a graph of the number of voice calls in progress at each instant of nodal time over the busy hour. Since the voice traffic component of the total nodal traffic load is identical for each of the five experimental runs, this graph remains constant and is applicable to all of the experiments. Figure 6.3, however, shows the number of data transactions in progress at each instant of nodal time over the busy hour in run #2B. Unlike voice calls, data transactions are very sensitive to many nodal parameters, and a different graph of data transactions in progress will occur for each run. The basic and advanced statistics included within Appendix III, as summarized in Table 6.2, provide sufficient data for the analysis of the following paragraphs.

6.3 DELAYS ENCOUNTERED IN THE NODE

The delays encountered by a unit of data (packet or segment) traversing the node under test can be considered as a composite of delays. For tandem packets, the total delay from complete input to complete output is composed of three types of well-defined delays: transmission delay, (in this case, for the trunk), queue delays; and processing delays. The following paragraphs assess the data presented in Table 6.2 in terms of these three types of delay.

6.3.1 TRANSMISSION DELAYS

Transmission delay accounts for the interval of time between the start of the first bit of a packet and the end of the last bit of that same packet. For the tandem case, the transmission delay then depends on the length of a packet and the effective rate of the trunk at the time of transmission. The mean and maximum transmission delays are a function of the available trunk data bandwidth which is determined entirely by the voice bandwidth in use at any given time. Since the mean and maximum voice trunk utilization are identical for all runs, this computation of transmission delays applies equally to each of the five experiments.

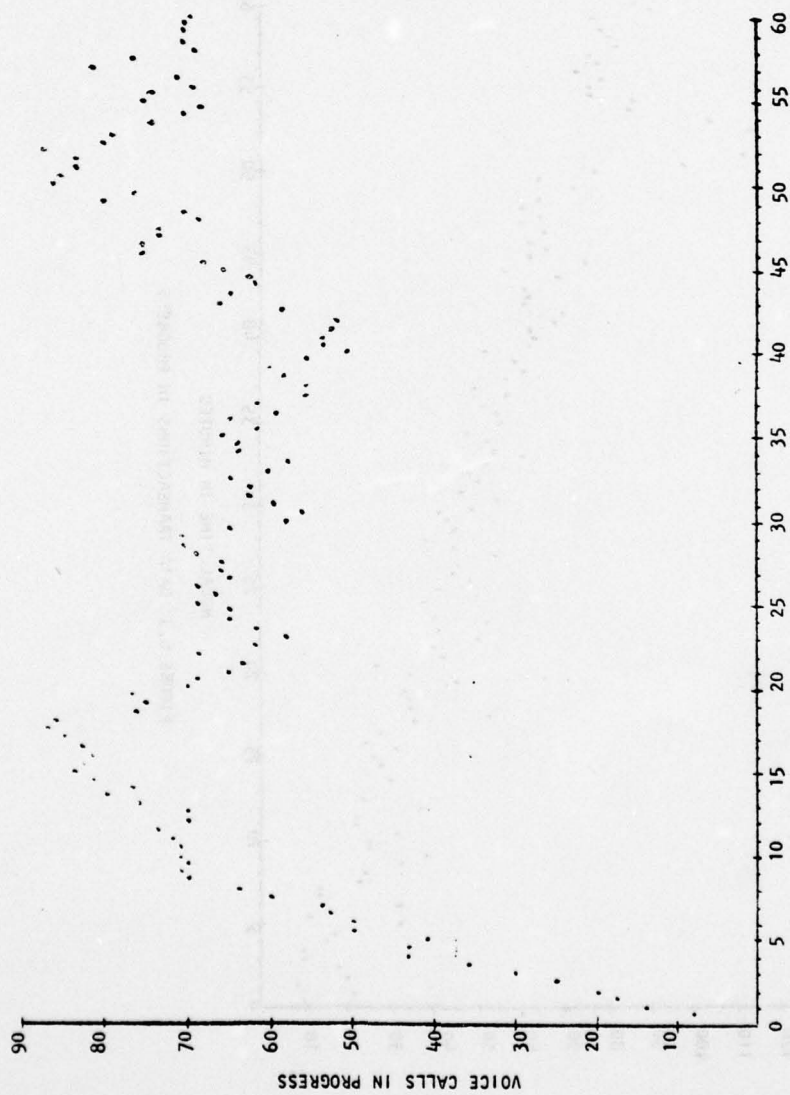


FIGURE 6.2 VOICE CALLS IN PROGRESS

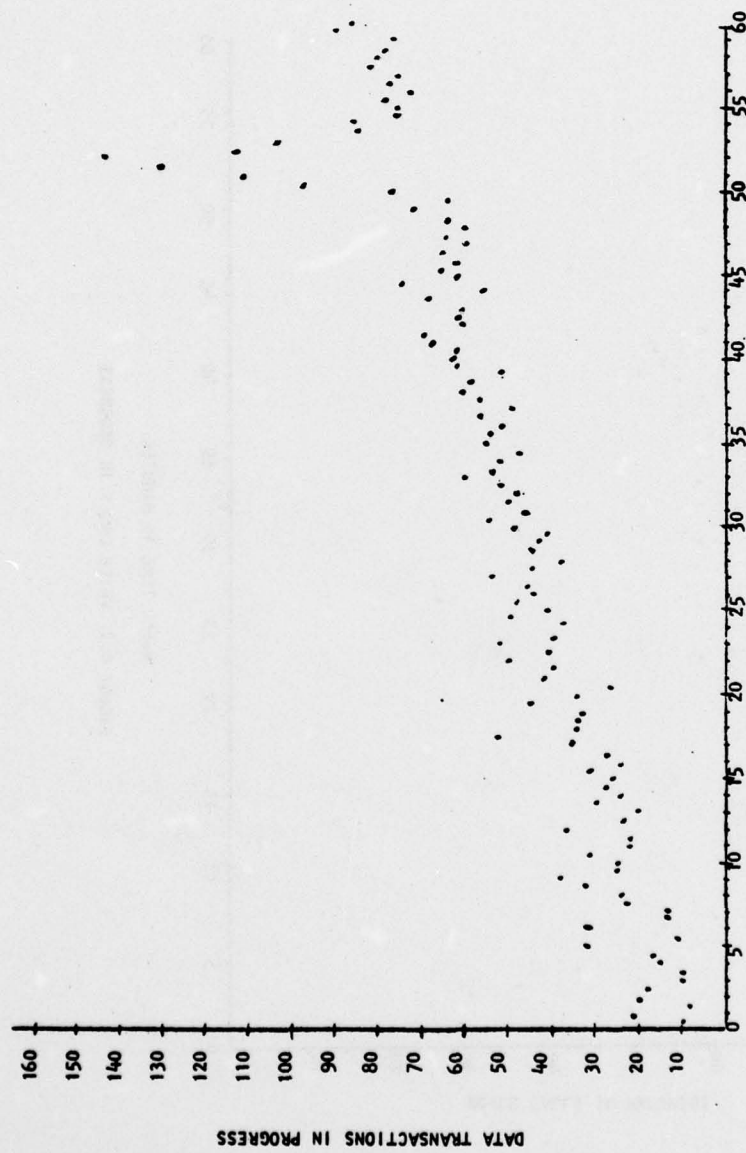


FIGURE 6.3 DATA TRANSACTIONS IN PROGRESS

From Table 6.2, the mean voice trunk utilization (\overline{VTU}) is 67.3% while the maximum (\hat{VTU}) is 91.2%. Thus, the mean available data bandwidth (\overline{ADB}) is the product of the trunk rate (T_R) and the difference between 100% and the mean voice trunk utilization. Symbolically,

$$\overline{ADB} = \frac{(T_R)(100\% - \overline{VTU})}{100\%}$$

For a T1 carrier rate, this equation becomes:

$$\overline{ADB} = \frac{(1.544 \times 10^6 \text{ bits/sec})(100\% - 67.3\%)}{100\%}$$

$$\overline{ADB} = (1.544 \times 10^6 \text{ bits/sec})(.327)$$

$$\overline{ADB} = 504,888 \text{ bits/sec}$$

A full packet contains 2048 bits and incurs a mean transmission delay (\overline{TD}_F) given by:

$$\overline{TD}_F = \frac{2048}{\overline{ADB}} = \frac{2048}{504888} \approx 4 \text{ milliseconds}$$

A control packet contains 144 bits and incurs a mean transmission delay (\overline{TD}_C) given by:

$$\overline{TD}_C = \frac{144}{\overline{ADB}} = \frac{144}{504888} \approx 0.29 \text{ milliseconds.}$$

Now, the minimum available data bandwidth (\hat{ADB}) is the product of the trunk rate (T_R) and the difference between 100% and the maximum voice trunk utilization (\hat{VTU}).

Symbolically,

$$\hat{ADB} = \frac{(T_R)(100\% - \hat{VTU})}{100\%}$$

For a T1 carrier rate, this equation becomes:

$$\hat{A}_{DB} = \frac{(1.544 \times 10^6 \text{ bits/sec})(100\% - 91.2\%)}{100\%}$$

$$\hat{A}_{DB} = (1.544 \times 10^6 \text{ bits/sec})(.088)$$

$$\hat{A}_{DB} = 135,872 \text{ bits/sec}$$

A full packet contains 2048 bits and incurs a maximum transmission delay (\hat{T}_{DF}) given by:

$$\hat{T}_{DF} = \frac{2048}{\hat{A}_{DB}} = \frac{2048}{135872} \approx 15 \text{ milliseconds}$$

A control packet contains 144 bits and incurs a maximum transmission delay (\hat{T}_{DC}) given by:

$$\hat{T}_{DC} = \frac{144}{\hat{A}_{DB}} = \frac{144}{135872} \approx 1 \text{ millisecond}$$

To summarize, the mean transmission delay for full packets is about 4 milliseconds while the mean for control packets is roughly 1/4 millisecond. The maximum transmission delay for full packets is approximately 15 milliseconds while the maximum for control packets is around 1 millisecond.

6.3.2 TRUNK QUEUE DELAYS

The time spent in queue waiting for service can be readily obtained for the tandem case since the node simulator monitors the trunk transmit queue and the four backlog queues. Unlike the transmission delays computed in the previous paragraph, these trunk queue delays are dependent on packet arrival rates and are, therefore, different for each experimental run. For the next tandem control packet, the mean time in queue can be obtained by a simple technique. First, assume that the transmit queue holds an equal number of full packets and control packets, on the average. (This is clearly indicated by consideration of available queue size in bytes of data.) Suppose that the mean number of items on the transmit queue is \bar{X} and the mean number of items

on the control queue is \bar{Y} . Then the mean time to transmit the items on the transmit and control queues and begin service of the next control packet (\overline{TW}_c) is given by:

$$\overline{TW}_c = (\frac{1}{2}) (\bar{X}) (\overline{TD}_F) + (\frac{1}{2}) (\bar{X}) (\overline{TD}_c) + (\bar{Y}) (\overline{TD}_c)$$

$$\overline{TW}_c = \frac{1}{2} \bar{X} (\overline{TD}_F) + (\frac{1}{2} \bar{X} + \bar{Y}) (\overline{TD}_c)$$

The quantities \overline{TD}_F , the mean transmission delay for a full packet, and \overline{TD}_c , the mean transmission delay for a control packet, were computed in the preceding paragraph. Similarly, suppose that the maximum number of items on the transmit queue is \hat{X} and the maximum number of items on the control queue is \hat{Y} . Then, under the same assumption as above, the maximum time to transmit the items on the transmit and control queues and begin service of the next control packet (\hat{TW}_c) is given by:

$$\hat{TW}_c = \frac{1}{2} \hat{X} (\hat{TD}_F) + (\frac{1}{2}) (\hat{X}) (\hat{TD}_c) + (\hat{Y}) (\hat{TD}_c)$$

$$\hat{TW}_c = \frac{1}{2} \hat{X} (\hat{TD}_F) + (\frac{1}{2} \hat{X} + \hat{Y}) (\hat{TD}_c)$$

The quantities \hat{TD}_F , the maximum transmission delay for a full packet, and \hat{TD}_c , the maximum transmission delay for a control packet, were computed in the preceding paragraph. The computation of the mean and maximum queue delay for tandem control packets is summarized in Table 6.3. The mean number of items on the transmit queue \bar{X} , the mean number of items on the control queue \bar{Y} , the maximum number of items on the transmit queue \hat{X} , and the maximum number of items on the control queue \hat{Y} were obtained from Table 6.2. Thus, Table 6.3 presents the mean trunk queue delay for tandem control packets \overline{TW}_c and the maximum trunk queue delay for tandem control packets \hat{TW}_c for each of the five experimental runs.

Finding the queue delays for other than control packets merely involves adding to the \overline{TW}_c value the delay delta measured for

RUN #	\bar{X}	\bar{Y}	TD_F (ms)	TD_C (ms)	TW_C (ms)	\hat{X}	\hat{Y}	\hat{TD}_F (ms)	\hat{TD}_C (ms)	\hat{TW}_C (ms)
1	1.02	.26	4	.29	2.26	7	5	15	1	61
2A	3.15	.70	4	.29	6.96	7	7	15	1	63
2B	1.95	.32	4	.29	4.28	7	4	15	1	60
2C	1.32	.10	4	.29	2.86	7	3	15	1	56
3	3.40	.30	4	.29	7.38	7	3	15	1	59

TABLE 6.3 COMPUTATION OF QUEUE DELAYS

each data priority then subtracting the difference in the transmission delay of a data packet as contrasted to a control packet. Table 6.4 represents the computed mean and maximum trunk queue waiting times for each priority. Due to limitations of the experimental setup correct maximum delay values were not obtained for runs 2A and 3. This can be corrected in future runs. Indications are that priority three maximum waiting times for run #3 could become very large (based on examination of throughput available versus transactions prematurely terminated).

6.3.3 PROCESSING DELAYS

Processing delays are the aggregate sum of all time spent in the node under test not directly attributed to transmission delay or trunk queue delay. For the tandem control case, processing delays are easily computed. The mean processing delay (\overline{PD}_c) for control packets is the difference between the mean delay in the node (\overline{ND}_c) and the sum of the mean transmission and mean queue delays.

Symbolically,

$$\overline{PD}_c = \overline{ND}_c - (\overline{TD}_c + \overline{TW}_c).$$

Similarly, the maximum processing delay (\widehat{PD}_c) for control packets is the difference between the maximum delay in the node (\widehat{ND}_c) and the sum of the maximum transmission and maximum queue delays. Thus,

$$\widehat{PD}_c = \widehat{ND}_c - (\widehat{TD}_c + \widehat{TW}_c)$$

The quantities \overline{TD}_c , \overline{TW}_c , \widehat{TD}_c , and \widehat{TW}_c were computed in preceding paragraphs. The mean delay in node \overline{ND}_c and the maximum delay in node \widehat{ND}_c for tandem control packets was obtained from Table 6.2

The result of the computation of the mean processing delay \overline{PD}_c and the maximum processing delay \widehat{PD}_c for each of the five experimental runs are summarized in Table 6.5. Dashes indicate a saturation

	TW _C (ms)		TW ₁ (ms)		TW ₂ (ms)		TW ₃ (ms)	
	MEAN	MAX	MEAN	MAX	MEAN	MAX	MEAN	MAX
1	2.26	61	2.76	92	3.16	127	3.56	130
2A	6.96	63	11.16	-	13.36	-	22.96	-
2B	4.28	60	4.98	98	6.18	154	7.08	1439
2C	2.86	56	3.46	96	4.16	786	3.86	1818
3	7.38	59	11.68	-	12.08	-	12.38	-

TABLE 6.4 QUEUE WAIT TIMES

condition pertaining to buffer utilization discussed in that paragraph. The occurrence of this condition, however, skews the maximum observed delays and results in dubious statistics, and is thus omitted. Since no processing priorities were employed in the node processing times computed for control packets (Table 6.5) are representative of processing times for all packets handled by the node.

6.3.4 IMPACT OF PRIORITY HANDLING

The results of Table 6.2 and 6.4 clearly indicate the effect of transmission priority handling on both the mean and the maximum tandem packet delay for each of the runs. For run #1, 2B and 2C the mean tandem packet delay for control packets is lower than that incurred by the other three priorities; priority 1, 2 and 3 packets tend to be very close in mean tandem packet delay. The maximum tandem packet delay shows a marked distinction among the control, and priority 1-3 categories with priority 3 showing long delays at the higher traffic volumes. The maximum tandem packet delay information for run #2A and run # 3 is skewed by the occurrence of the condition mentioned in the preceding paragraph, and is thus omitted from further consideration. For run #1, 2B, and 2C, the reason for this observed impact of priority handling on tandem packet delay is indicated in the Table 6.2 listing of queue sizes. The mean queue size for each of the four priority backlog queues (control and priority 1-3) is much less than unity for each of these experimental runs. Hence, there was not a great deal of interference as the four priorities contended for service on the output trunk, and the mean tandem packet delays are approximately the same for all priorities. However, the maximum queue sizes given for each of the four priority backlog queues are much larger than unity for each of these experimental runs. As a consequence, there is a great deal of interference as the four priorities contend for service on the output trunk, and the maximum tandem packet delays vary over a wide range based on packet priority.

AD-A049 619

RCA CORP CAMDEN N J

F/G 17/2

STUDIES FOR DEVELOPMENT OF A UNIFIED NODE EMPLOYING DYNAMIC CHA--ETC(U)

DEC 77 B PATRUSKY, K BODZIOCH, R CHAN

F30602-75-C-0109

UNCLASSIFIED

RADC-TR-77-380

NL

4 of 5
AD
A049619



RUN #	\overline{ND}_C (ms)	\overline{TD}_C (ms)	\overline{TW}_C (ms)	\overline{PD}_C (ms)	\hat{ND}_C (ms)	\hat{TD}_C (ms)	\hat{TW}_C (ms)	\hat{PD}_C (ms)
1	27.8	.29	2.26	25.3	153	1	61	91
2A	50.1	.29	6.96	42.9	-	1	63	-
2B	16.3	.29	4.28	11.7	102	1	60	41
2C	5.9	.29	2.86	2.75	97	1	56	40
3	9.5	.29	7.38	1.83	-	1	59	-

TABLE 6.5 COMPUTATION OF PROCESSING DELAYS

6.4 BUFFER UTILIZATION

The utilization of nodal buffers during the five experimental runs is summarized in Table 6.2, and presented in considerably greater detail in the advanced statistics reports generated by the data reduction program which are included in Appendix III. The mean number of buffers in use for all five experimental runs seems to be relatively small in comparison to the number of data sources and data traffic volume. The maximum number of buffers in use for runs #1, 2B, and 2C also appear to be relatively small which indicates high buffer turnover. However, it should be pointed out that buffer utilization was on the rise at those points in runs 2A and 3 where the simulation reaches saturation.

6.4.1 PREDETERMINED THRESHOLD THROTTLING

During one of the preliminary experimental runs using the same data traffic load as run #1 (12,500 transactions/hour), the maximum number of nodal buffers was set at 80 with the same throttling thresholds as specified in the five experimental runs summarized in Table 6.2. Block-by-block, connection-per-block, flash only, and trunk input barring were all observed in addition to normal operation. Acceptance delays increased, and the number of active data transactions quickly built up to the simulator limit of 384. At this point, the network traffic generator began discarding the overflow transactions. The predetermined nodal buffering thresholds were not exceeded during the five experimental runs since the network simulator resources entered saturation before the throttling rules incorporated in the node were invoked. Consequently, throttling was not observed in any of the five full experimental runs.

6.4.2 NATURAL THROTTLING

An interesting situation developed during run #2A and run #3. Again, the number of active data transactions in progress reached the network simulation limit of 384, and the traffic generator

began to discard the overflow. However, in these two cases the reason for this situation was indicative of a fundamental characteristic of nodal performance. The nodal software program is capable of processing a certain number of transactions per second, independent of the buffering resources available. When this limit is reached, "natural throttling" occurs. Control response times lengthen and queues grow causing delivery times to increase. At this point, since transactions already in progress are taking longer to complete, new transactions already in progress are taking longer to complete, new transactions supplied by the traffic input tape continually build up in the network simulation until the limit of 384 is reached. This situation causes overflow transactions to be discarded and prevents buffer utilization in the node from increasing to the point where the predetermined throttling thresholds would be exceeded resulting in the invocation of nodal throttling rules. If the simulation possessed more buffering and the node less buffering, this situation would be prevented. However, the occurrence of natural throttling did skew the results reported by data reduction for run #2A and run #3. As illustrated by consideration of run #2B and run #2C, increasing the effective processor speed (time scale factor) eliminated throttling caused by excessive processing delays.

6.5 TRUNK UTILIZATION

The single trunk contained in the ADPT node simulator configuration on which the five experiments were conducted employed dynamic channel allocation. Of the 1544 bits of trunk bandwidth available each millisecond, a maximum of 1408 were devoted to voice traffic, with the remaining 136 bits per millisecond dedicated to data traffic. Any portion of the voice trunk bandwidth not used for voice traffic was available for use by data traffic, in effect, supplementing the dedicated data trunk bandwidth.

6.5.1 VOICE UTILIZATION

Table 6.2 summarizes the voice utilization of trunk bandwidth for each of the five experimental runs while a detailed second-by-second account of voice trunk utilization is contained in the system edit printouts (which were not included in Appendix III because

they are bulky). The mean voice utilization is 67.3% and the maximum voice utilization is 91.2% for all five experimental runs.

6.5.2 COMBINED TRUNK UTILIZATION

The mean and maximum trunk utilization for both input and output are given in Table 6.2 for all five experimental runs, while detailed second-by-second accounts of input and output trunk utilization for each of these runs is contained in the system edit printouts. The combined voice and data trunk utilization observed during the five experimental runs varied from a mean of 71.7% to a maximum of 100%.

It was interesting to observe that when data traffic was very light (run 1) the data did not queue sufficiently to fully utilize the trunk. However, 100% trunk utilization was observed in later runs where more data traffic was introduced.

6.6 THRUPUT

The mean and maximum thruput in terms of both input and output are summarized in Table 6.2 for each of the five experimental runs, while detailed second-by-second accounts of input and output in packets per second for each run are contained in the thruput edit printouts (which were not included in Appendix III because of space limitations). The maximum observed thruput during these five experimental runs occurred during run #3 and approached 1000 packets per second. Since a processor scale factor of 4 was used this indicated that a processor having the speed of the last processor is capable of a thruput of 250 packets per second which was consistent with the thruputs achieved in run 1 where the process scale factor of 1 was employed.

6.7 CONCLUSIONS

A key result of the relatively few experiments run was that the results were consistent with each other. Trunk utilizations

cross-correlated correctly with thruput measurements. Queue sizes correlated correctly with measured delays. Delays increased with traffic load and indicated priority impacts distinctly. It can be stated with high confidence that the simulation is operating correctly.

Mean trunk delays were found to be of the same magnitude as that predicted by the analytic model. (Appendix IV).

In most of the runs where the simulator did not saturate the average data traffic introduced was only slightly more than the dedicated bandwidth for data. Under these conditions both the analytic model and experiments both indicated mean delays of about 5 milliseconds. In the case of run 3, 16 erlangs of data traffic was introduced with 8 dedicated data channels provided. The measured mean delay was only 7 milliseconds while the analytic model indicated an expected delay of about 15 milliseconds. In this run the simulation saturated, eliminating the worst delay peaks, thus indicating a more favorable mean than should be indicated for this run. Since the duration of these delay peaks were small and represented a small percentage of the total run data it is doubtful whether the actual mean would exceed 15 milliseconds for run 3.

The conclusion is that mean delays are reasonable values which can satisfy delivery times suggested for handling of bulk transactions which employ trunk bandwidth also utilized by voice traffic.

However, there is a pronounced tendency for very large differentials to exist between mean and max. delay values. Delays well over a second are likely to occur during voice blocking intervals for those data priorities which are allocated for occupying voice bandwidth. The analytic model

primarily developed mean values. Maximum wait times developed in the analysis assumed an exponential distribution of wait times. In actuality discrete huge peaks occur during the relatively infrequent occasions of maximum voice occupancy. In fact, the initial experiments were not adjusted properly to avoid simulation

saturation, thus the largest peaks which would naturally occur in run 3 were obscured.

Examination of the voice peak characteristics of each run indicated that one voice peak lasted for twenty seconds. During this period priority III traffic must be throttled else huge backlog queues will result. A tentative conclusion is that dedicated data channels must be employed for handling the interactive traffic and voice/data contention channels can be used for bulk and narrative data.

The throughput achieved using the R-100 was 250 packets/second. Actually, this value is not as fast as the number suggests. Half of the packets are control packets and each packet both enters and exits the node and is therefore counted twice. Actual packet handling capacity of the node using the lab processor (R-100) was slightly greater than 60 packets per second (although higher processing capability can be simulated via the scaling parameter). If future experiments are conducted an attempt could be made to improve the node throughput via a restructuring of software priorities. Since the lab processor is typical of the mini class of processor and since large nodes may have to handle close to a thousand packets per second (2048 bit packet) the future node may either have to employ many parallel processors, longer packets, more front end processing, a more sophisticated software structure or a combination of all of the above.

Priority queues on output are useful to assure fast handling of control and interactive traffic. The results so far suggest that a three priority structure is adequate.

Priority I - Control

Priority II - I/A Traffic

Priority III - Bulk and Narrative

One of the key recommendations is to perform more tests using this very powerful tool to develop and validate candidate node architecture designs to improve the node software design as regards packet throughput and to perform parametric analysis varying blocking probabilities, throttling parameters, routing algorithms, etc. and to develop a larger data base for analysis of nodal behavior. See Section 9.

7.0 FUTURE NODE ARCHITECTURE - HARDWARE

7.1 INTRODUCTION

The purpose of this study is to trade off various architecture candidates to define the nodal architecture of a future switching node.

The key differences in performance requirements between switch nodes of the 1980's and present nodes are:

- a) Data traffic is predicted to increase substantially as future computer systems communicate via common user networks
- b) The large volume of short interactive transactions typically associated with man-computer and computer-computer communications requires short connect times in order to communicate efficiently.
- c) Voice communications will tend to be digitized for purposes of encryption and to insure good quality links irrespective of communication path length or variety of circuits employed. This will result in a large increase in synchronous data traffic volume. Furthermore, it is likely that a common user network will have to accommodate multiple synchronous data rates.

The objective of this study is, therefore, to analyze the requirements of the future node and perform the necessary technical analysis/evaluation to define a nodal architecture capable of providing the user services required, and capable of handling the anticipated volume of voice, data and record traffic in a cost effective manner. Key factors to be considered in the development of a node architecture is the need for flexibility and modularity.

Flexibility will provide for anticipation and accommodation of predictable and unpredicted requirement changes.

Modularity is a key design requirement because nodes in a large network will vary considerably in capacity and type of traffic to be accommodated. Modularity will also permit wider application of the node architecture. Furthermore, Air Force Base switches are likely to be considerably smaller than those switches acting as the backbone of the future DCS. This consideration further emphasizes the importance of the modularity requirement.

The study is organized in the following manner. An analysis of the future DCS node requirements develops the design objectives/goals. Candidate hardware architectures are then synthesized. Evaluation criteria are then developed and the candidate architectures are then synthesized. Evaluation criteria are then developed and the candidate architectures are weighed against the criteria established. Finally, the recommended architecture is defined in greater detail. It is noted that nodal architecture was examined separately from the standpoint of hardware and software architectures. It was, therefore, necessary to define hardware/software boundaries when candidates were synthesized. This section concentrates on the hardware architecture. The following section concentrates on the software architecture.

7.2 NODE REQUIREMENTS

7.2.1 NODE TRAFFIC

The future node must accommodate both digital voice and data access lines and each node in the network will incorporate both access and tandem functions.

Three classes of digital traffic will be accommodated:

Class I - real-time traffic that is continuously transmitted, started, such as; voice, video, and certain forms of facsimile traffic.

Class II - the general class of store and forward traffic such as high-speed interactive traffic now being handled by packet switching, and the slower narrative/message traffic.

Class III - bulk data transfers such as data base transfers between computers with the essential feature that the flow of traffic can be controlled by the communication system.

7.2.2 SWITCH FUNCTIONS

It is projected that the future node will accommodate present day voice and narrative record traffic and also offer data packet switching service to accommodate the handling of short interactive traffic. The following is a summary of switching capabilities required of the future node.

1. Circuit Switching - The ability to provide call connections to provide transparent voice (analog or digital) and data communications for extended holding times.
2. Store and Forward Message Switching - Primarily for narrative/record traffic, this service provides for storing, and full accountability for delivery (following acceptance) of complete variable length message traffic.

3. Packet Switching - This new service provides for data transmission of packets (messages or parts of messages). The packet is an independent entity containing a quantum of data and preceded by a header containing all necessary control and routing information. The network is accountable only for individual packet acceptance and delivery. Data transfer is near real time in that packets are not delivered to the network unless they can be accepted by receiving terminals. Thus throttling occurs at source terminals. Different packets associated with a single message can reach their common destination via independent routes.
4. Bulk Data Service - It is possible that a special service could be offered for handling bulk data traffic wherein semipermanent links could be established for transfer of bulk traffic. These links would differ from true circuit switched links in that they would be dynamically interruptible, that is, they would be throttled temporarily if Class I or Class II data required the available bandwidth, but would automatically be reestablished as bandwidth becomes available. The need for this service is not yet clearly established since Class III traffic could also be handled as either a pre-emptible Class I call or as low priority Class II packets. A separate study assesses the complexity of offering a special bulk handling service. (See section 2.2)

7.2.3 NODE INTERFACES

7.2.3.1 VOICE SUBSCRIBER LINES

The following voice subscribers would have to be accommodated:

- a) Standard Analog Phones

- b) AUTOVON Switch trunks
- c) TRITAC and DAX Switch trunks
- d) Digital Non Secure and Secure Phones (DSVT, DNVN)

7.2.3.2 SYNCHRONOUS DATA SOURCES

- a) FAX
- b) Video
- c) On Line Graphics (i.e., Light Pen Terminals, Optical Readers)

7.2.3.3 ASYNCHRONOUS DATA SOURCES

- a) Mode IIA data (i.e. TTY)
- b) Mode I data (i.e. Autodin, DSTE)
- c) Mode VI (ADCCP) data (i.e. Intelligent terminals, host computers)

7.2.4 INTERNODE TRUNKS

It is planned to incorporate multiplexing within the future node in order to take advantage of the lower transmission costs involved in leasing or buying multiplexed trunks rather than individual lines. Available line rates currently offered by the Bell system include 56 KBS links and T-1 links. The T-1 rate is 1.544 MBS, however, 8 KBS is reserved for special framing bits thus 1.536 MBS capacity remains for information. Since use of the T-1 link is economical, the T-1 rate will be utilized where possible and will be the basis for current planning, however, the node design must not preclude usage of lower data rate trunks. It is assumed that rates higher than T-1 can be handled by separate multiplexers.

7.2.5 NODE SIZE

Node size is a function of the amount of traffic to be accommodated and the number and type of node access lines. Since the size estimates are for nodes to be operational in 1985 and beyond, predictions of size requirements were obtained from planning documents provided by DCA and predicted growth rates of existing traffic.

Table 7-1 represents a distillation of size requirements based on network data presented in the DOD and DCS planning documents referenced

Appendix A presents the rationale used to develop these traffic handling requirements. The small node is representative of the 25% smallest nodes in the network and can be considered to be an appropriate design size for the smallest processing module. The super nodes are those located at such concentrated population centers as Arlington and Mosely. Since these nodes have much more load than other network nodes, figure designs could consider splitting these sites into two large nodes, thus avoiding the need to design a super node. The "large" node represents the largest nodes left when the super sites are eliminated. A fairly large number of sites will have to accommodate this load, especially if super nodes are split into two or more large nodes. The large node requirements would then be treated as the largest single node design requirement (if splitting up the super nodes proves feasible).

Several design considerations are worth special mention at this point. The super node must handle approximately 16 times the traffic of the small node and the large node approximately 9 times. This factor is an important design consideration pertaining to modularity.

It is noted that the data throughput requirement is compatible with present day processor bulk memories. For example, a memory with 750 nsec access time can accommodate 10.66 MBS throughput with byte access and 2132 MBS throughput with half word (two bytes) access. Handling of the digital voice load, however, will either require use of super high speed buffer memories or parallel access techniques.

It is also noted that a memory size in the neighborhood of a megabyte is required for super nodes. If shared or common memory is employed for traffic buffering the CPU's which will access the shared memory should have capability to directly access the entire memory. Present data microprocessors do not offer

TABLE 7-1 NODE SIZES

<u>VOICE TRAFFIC LOAD</u>				
	<u>Small</u>	<u>Average</u>	<u>Large</u>	<u>Super</u>
Load	70 E	235 E	625 E	1125 E
Lines	200	600	1200	2400
Trunks	100	280	670	1160
Avg. Thruput	4.48 MBS	15 MBS	40 MBS	72 MBS
Max.	6.4 MBS	18 MBS	42.8 MBS	74.2 MBS
<u>DATA TRAFFIC LOAD</u>				
	<u>Small</u>	<u>Average</u>	<u>Large</u>	<u>Super</u>
Load	80 P/S 160 KBS	270 P/S 540 KBS	720 P/S 1.44 MBS	1300 P/S 2.6 MBS
Avg. Thruput	320 KBS	1.08 MBS	2.88 MBS	5.2 MBS
Peak Second	736 KBS	2.48 MBS	6.62 MBS	11.96 MBS
Lines	70	210	420	840
Buffers for Pkts.	42,500 B	130,000 B	346,666	626,000 B
Tables	2,000 B	6,000 B	12,000	24,000 B
Prog. Size	12,000 B	12,000 B	12,000	12,000 B

large addressing capacity and only some of the present day minicomputers offer the required addressing capability. This limitation can be circumvented by introducing special hardware for fast data transfer between CPU memory and a large shared memory.

7.2.6 PERFORMANCE REQUIREMENTS

Voice performance requirements are defined by "grade of service" and "call connect time delays". A total network grade of service defined as call blocking probability shall be 7%. Blocking probability at any network node is a function of network interconnectivity but will probably range between 1% and 5%.

Present day call connect times can be as long as 10 seconds, however, employment of common channel signal/supervision techniques should permit considerable improvement in this area. A reasonable design goal appears to be 2 seconds worst case connect time, including one or two satellite hops.

Packet data calls are delayed, rather than blocked. Delays specified are "Acceptance" and "Delivery" delays. Acceptance delay is a measure of quality of service and is defined as the percentage of transaction first segments for each traffic acceptance category that wait more than a specified time before being accepted for delivery by the network.

Delivery delay is defined as the sum of acceptance delay, transmission delay and message transfer delay. Tables 7-2 and 7-3 are delay values as specified in the Autodin II specification. These values will serve as guidance for developing node design parameters.

7.3 FUNCTIONAL OVERVIEW

Having defined the basic functional and performance requirements, it is now possible to provide a functional overview of the node and a functional block diagram. Fig. 7-1 presents a simplified

TABLE 7-2 AUTODIN II ACCEPTANCE DELAY

<u>CRITICALITY</u>	<u>CATEGORY</u>	<u>QOS SPECIFICATION/ ACCEPTANCE DELAY</u>
Flash Override (Y&W)	I	Nonblocking
Flash (Z)	I	Nonblocking
Immediate (O)	II	Blocking - $P(t > \epsilon) \leq .01$ $P(t > 5 \text{ sec}^*) \leq .01$
Priority (P)	III	Blocking - $P(t > \epsilon) \leq .01$ $P(t > 10 \text{ sec}^*) \leq .01$
Routine (R)	IV	Blocking - $P(t > \epsilon) \leq .01$ $P(t > 30 \text{ sec}^*) \leq .01$

TABLE 7-3 AUTODIN II DELIVERY DELAY

<u>MAIN CATEGORY</u>	<u>TYPE OF TRANSACTION</u> <u>SUBCATEGORY</u>	<u>DELIVERY DELAY</u>	
		<u>MEAN</u>	<u>MAX.</u>
Interactive	Human Interaction	1 sec	2 sec
	Alarms/Status	1 sec	3 sec
	Indicators		
	Monitoring/Telemetry	0.3 sec	1 sec
Query/Response	--	36 sec	2 min
Narrative/Record*	Flash/Flash override	4 min	10 min
	Immediate	6 min	30 min
	Priority	9 min	3 hrs
	Routine	16 min	6 hrs
Bulk 1	--	5 min	20 min
Bulk 2	--	4 hrs	12 hrs

*Includes AUTODIN I Transmission Times

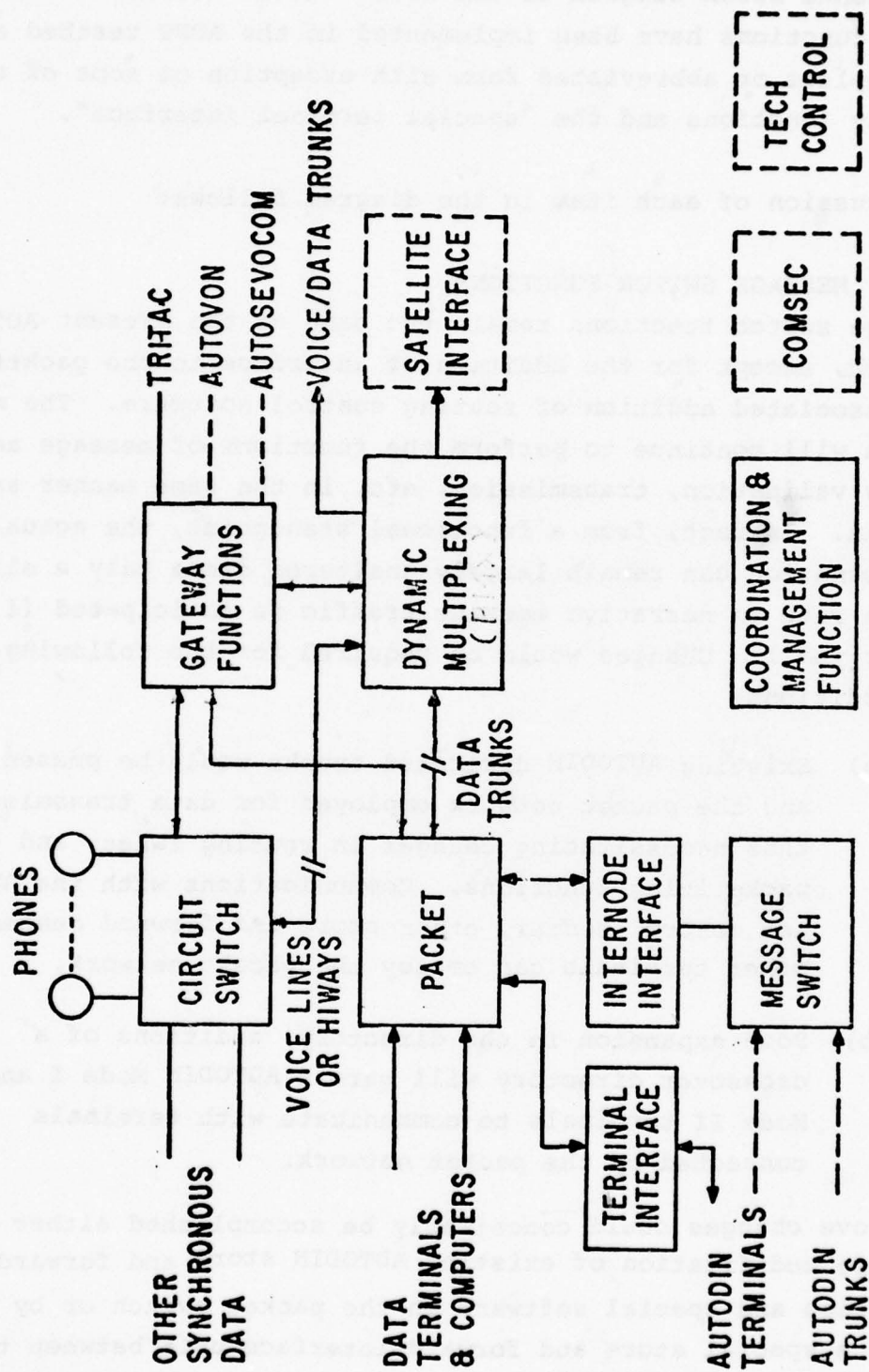


FIGURE 7-1 UNIFIED NODE FUNCTIONAL BLOCK DIAGRAM

functional block diagram of the node. It is noted that all these functions have been implemented in the ADPT testbed either in complete or abbreviated form with exception of some of the gateway functions and the "special terminal interface".

A discussion of each item in the diagram follows:

7.3.1 MESSAGE SWITCH FUNCTIONS

Message switch functions remain the same as the present AUTODIN network, except for the addition of interface to the packet network and associated addition of routing control software. The message switch will continue to perform the functions of message accumulation, filing validation, transmission, etc. in the same manner as in Autodin. In fact, from a functional standpoint, the actual AUTODIN implementation can remain largely unaltered since only a slight growth rate in narrative message traffic is anticipated (i.e. 2% per year). Changes would be required for the following alterations and additions:

- a) Existing AUTODIN dedicated trunks would be phased out and the packet network employed for data transmission thus necessitating changes in routing rules, and certain packetizing functions. Communications with the AUTODIN net control center, other store and forward centers and other terminals can employ the packet network.
- b) Some expansion in the directory, additions of a crossover directory will permit AUTODIN Mode I and Mode II terminals to communicate with terminals connected by the packet network.

The above changes could conceivably be accomplished either by software modification of existing AUTODIN store and forward facilities and special software in the packet switch or by interposing a special store and forward interface unit between the existing S & F function and the packet switch. In any case, this interface is highlighted because it represents a major change of the existing S & F implementation.

7.3.1.1 SPECIAL AUTODIN TERMINAL INTERFACE

Development of a front end processor unit which interfaces AUTODIN Mode I and Mode II terminals and the packet switch appears highly desirable. The DCS plan for backbone nodes suggests that the future backbone network will include 60-70 CONUS nodes. However, there is no requirement for significant expansion of existing Store and Forward (S&F) capability. Therefore, it is probably most realistic to integrate existing Autodin nodes into colocated unified nodes, and not include S&F capability in the remaining nodes.

Significant transmission cost savings should be possible if Autodin terminals located long distances away from the Autodin site and located near the more plentiful non S & F nodes can access the Autodin site via the packet network. Further savings can be realized since a single packet physical link can handle multiple virtual connections thus permitting the design of the special access terminal concentrator which either can be remoted or colocated.

7.3.2 CIRCUIT SWITCH

The key changes anticipated in the circuit switch and associated functions are:

7.3.2.1 GATEWAY FUNCTIONS

Signalling and Supervision interfaces with other networks will probably employ CCIS. CCIS formats may undergo standardization, however, special "gateway" functions will be required to permit appropriate interpretation of the individual signalling data. As unified nodes replace AUTOVON and AUTOSEVOCOM nodes the interim trunks to these networks will be phased out.

7.3.2.2 ACCESS INTERFACE AND MATRIX FUNCTIONS

Conversion to use of all digital phones will permit end to end encryption of voice links, will provide better quality long distance communications, will permit implementation using low cost digital matrices, and will permit digital multiplexing of voice and data traffic on transmission links.

The digital circuit switch provides the means for employing the common user network to handle the following types of synchronous data calls.

- a) FAX
- b) Video
- c) Interactive Graphics
- d) Dialed Data Service

A multiplicity of digital data rates must be accommodated to handle different voice rates and synchronous data rates and to anticipate the eventual reduction in voice rates as analog/digital conversion techniques improve.

An analog matrix may be required with crossover links to digital matrices. It is likely that analog phones will be slowly phased out, however, a long interim conversion period is likely. Crossover links will permit intercommunications between analog and non-secure digital phones and between non-secure digital phones employing different data rates. It is assumed, however, that end to end encryption will be restricted between compatible terminals. It is not within the scope of this study to determine whether existing analog matrices employed in AUTOVON and AUTOSEVOCOM should be used or whether it is more cost effective to replace the analog switch matrix with solid state matrices (the above statement assumes that unified nodes will replace AUTOVON and AUTOSEVOCOM nodes).

7.3.2.3 CONTROLLERS, SCANNER, TONE GENERATORS, AND RECEIVER SENDERS

These functions are generally familiar to circuit switch designers. Scanners detect On Hook/Off Hook conditions and relay the hook status to the controller.

Digital Receivers detect signalling digits entered by the user and relay the signalling data to the controller.

Tone Generators provide, for example, busy, ring, and pre-empt tones to the user.

The Controller gathers signalling information and controls switching user phones and terminals to circuits, digit receivers, and tone generators as required. The Controller will also coordinate internodal Signalling and Supervision by generating and receiving CCIS traffic.

The key new considerations are:

- a) The need for digital signalling and supervision standards to permit the use of common digit receivers and digital tone generators for different types of digital phones.
- b) RCA has recommended that CCIS be accomplished employing control packets transmitted via the packet switch. This implies that the circuit switch controller must interface the packet switch function to provide for generation and reception of CCIS packets.
- c) It is planned to employ multiplexed trunks for internodal communications.

Dynamic Allocation requires that subscriber channels be reallocated each time a trunk call is added or deleted. The controller must interface the multiplex function to "program" the multiplex operation.

7.3.3 PACKET SWITCH

Figure 7-2 is an expansion of the packet switch functional block depicted in Figure 7-1.

The packet switch consists primarily of data block acceptance, validation, and routing functions. Each packet is a data block preceded by a header. The header contains the destination address which is used for routing. In a military switch the header also indicates priority, category and security. This data is used to establish processing and transmission priorities, and to prevent security violations.

Since packets have a finite maximum block length, data buffers can be preassigned to accept incoming packets. Control information for flow control, acknowledgements and other control is transferred between nodes via control packets which are shorter in length than maximum size data packets.

Messages are broken into segments for transmission and reassembled upon reception. Segments are converted to packets by the network node. In most instances, segment formatting and segment reassembly functions will be functions performed by network terminals (i.e. Host computers). Some nodes may be required to terminate ordinary message terminals directly. It is assumed that, in those instances, these functions will be accomplished by front end processors and will not impact the central packet switch implementation.

Segments and packets contain the same data, however, "segments" are data blocks transferred between terminals and a network node while "packets" are data blocks transferred between network nodes.

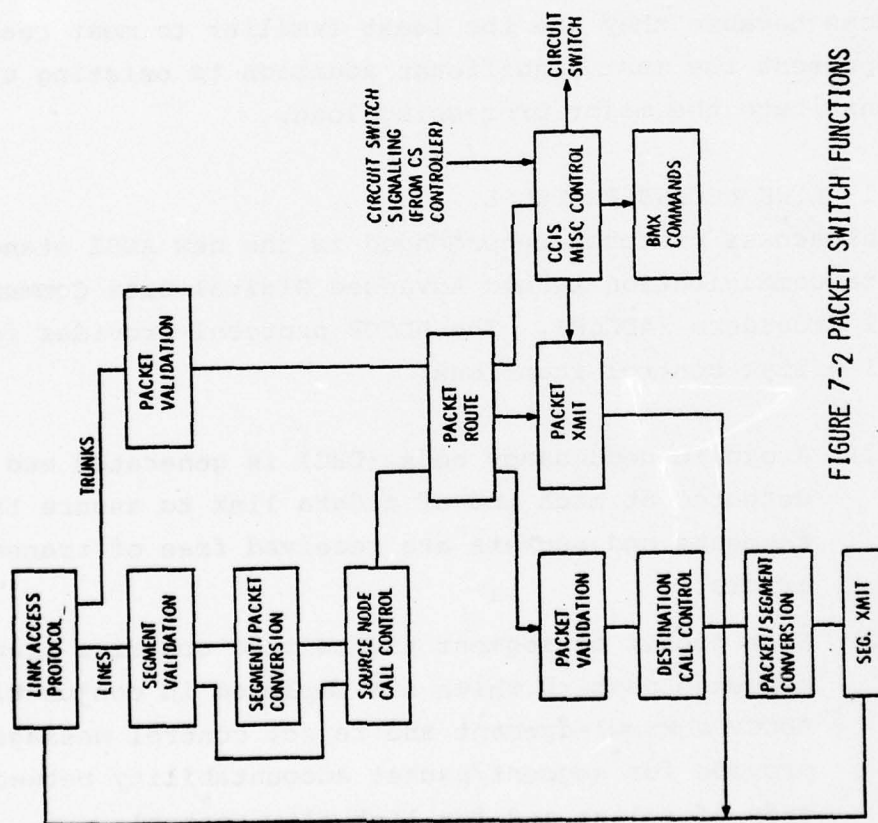


FIGURE 7-2 PACKET SWITCH FUNCTIONS

All packets have a common "header" format. Segments, however, have "leaders" which differ from "headers" in that they do not contain network control data, and have formats designed to be compatible with the data terminals. It is likely that a number of different segment leader formats will have to be accommodated by the node.

The source and destination network nodes will perform the function of segment/packet and packet/segment conversions.

Proceeding from this background discussion, let us now discuss the individual functions depicted in Figure 7-2.

Special emphasis is placed on discussing the packet switch functions because they are the least familiar to most reader, yet represent the most significant addition to existing systems and constitute the major processing load.

7.3.3.1 LINK ACCESS PROTOCOL

The link access protocol recommended is the new ANSI standard for data communication termed Advanced Digital Data Communication Control Procedure (ADCCP). The ADCCP protocol provides for the following link control functions:

- 1) A cyclic redundancy code (CRC) is generated and detected at each end of a data link to assure that segments and packets are received free of transmission errors.
- 2) Each packet or segment transmitted contains transmit sequence numbers which are employed in conjunction with ADCCP acknowledgement and reject control messages to provide for segment/packet accountability between ends of a link and for link flow control.

- 3) Flag characters envelope packets and segments to provide start and end packet markers. These flags are generated at the transmit end of a link and detected at the receive end.
- 4) Bits are "stuffed" at the send end of a link and "deleted" at the receive end to assure that flags and aborts bit configurations do not appear in the text of the transmitted data (all of these control characters described have six contiguous "ones"). Inserting a zero in text following five contiguous "ones" prevents these characters from appearing in text and permits text data to be transparent (no coding restrictions).

It is noted that functions 1, 3 and 4 involves single bit or single byte processing of a repetitive nature. Considering the extremely large data thruput requirements projected, use of front end hardware or firmware to perform these functions is mandatory. RCA has developed, during this program, a hardware device called an "Interactive Communication Channel" (ICC) which performs these and other functions to offload the software processing required.

7.3.3.2 SEGMENT VALIDATION

Incoming segment leaders are checked for validity. A valid segment does not contain any data fields which is invlaid (cannot be processed such as invalid destinations, precedence, category, etc.) Segment Validation is unique to terminal type and, therefore, lends itself to implementation employing "front end" processing.

7.3.3.3 SEGMENT/PACKET FORMAT CONVERSION

Incoming segment leaders are converted to packet headers. Packet headers combine node input data and leader data. Code conversion of leader data may be necessary (for example, numerics may be entered in ASCII code and be converted to Binary or BCD code).

If segment leaders are different in length than packet headers, additional processing is involved to close in or open up the header/data interface. RCA has found that considerable processing can be avoided by specifying leaders and headers to be the same size, using filler bits where necessary. Examination of proposed leader and header formats indicates that the filler overhead penalty is relatively small.

7.3.3.4 SOURCE NODE CALL CONTROL

This module of segment processing involves rejecting bad segments, processing special control segments, requesting new segments from source terminals, and maintaining accountability for correct sequencing of the segments of each message. This group of functions is specific to a data call or transaction and is to be performed by the source network node.

7.3.3.5 PACKET ROUTE

All received data, whether packets incoming from trunks or packetized segments incoming from lines, is routed to various output queues based on type of data in the packet and the destination address. Tandem data packet traffic is routed directly to trunk queues for packet transmit processing. Incoming non-CCIS data packets are routed to access line queues for validation and other processing and CCIS messages are routed to special trunk queues for special voice call processing.

7.3.3.6 PACKET VALIDATION

Packets received at a destination node are checked for valid line address and must have a security level or lower than the maximum acceptable level allowed for the destination line. Transmission control code restrictions must also be satisfied.

7.3.3.7 DESTINATION NODE CALL CONTROL

Packets received at a destination node are checked for correct sequencing within the data transaction and receipt is acknowledged to the source node either by assigning new buffer allocations or

by indicating that buffer allocations are not available. Timeouts check that ongoing data calls do not hang up in the system.

7.3.3.8 PACKET/SEGMENT CONVERSION

Processing may be required to convert the packet header into a "terminal unique" leader. Special processing may be required to code convert data if source and destination terminals employ different codes. Hopefully, future terminals will standardize on ASCII code.

7.3.3.9 SEGMENT TRANSMIT

Prepared segments will be queued for transmit according to priority. Segments will be moved to a common transmit queue when link sequence numbers can be assigned and will await transmission via interrupt processing. Transmitted packets will be moved to a "wait ack" queue to await link acknowledgement (part of link access protocol processing).

7.3.3.10 PACKET TRANSMIT

Packet transmit process is identical to segment transit process, but applies to trunks rather than access lines.

7.3.3.11 CCIS MESSAGE PROCESSING

Provides for establishing and disconnecting calls. Call initiate, complete, and release message are processed. Circuit switch signalling involving remote calls are also processed. The local circuit switch and trunk time slots (BMX switch) are appropriately controlled. CCIS message packets are generated and relayed, and acknowledged. Directories are maintained as required to analyze CCIS signalling data. Status and linking of calls in progress are maintained.

7.3.3.12 SUMMARY

A packet switch is really a form of switching employing data memory buffers and data processing. Acceptance and transmission

of a packet to and from buffers involves bit and byte processing which will be implemented by hardware or firmware.

7.4 PROCESSOR CANDIDATES

The processing system for the future node can theoretically range from a single processor to a large multiprocessor array. First a single processor system was examined and specialized. Input-Output units were developed to enhance the thruput of data packets. Subsequently, multiple processor and multiprocessor architectures were examined. A "multiple processor system" as used herein is a system utilizing a number of independent processors which are interconnected by buses or other means. A "multiprocessor system" as used herein is a system utilizing a number of processors under integrated control. Characteristically a multiprocessor system employs a single integrated operating system to control all hardware and software. Multiprocessors system is defined by ANSI as "A system employing two or more processing units under integrated control" (Vocabulary for Information Processing American National Standard X3.12-1970). A more detailed and descriptive multiprocessor definition is stated as follows:

- 1) There are two or more CPU's
- 2) Main Processor memory is shared and accessible by all processors (although processors may also have private memory).
- 3) I/O devices are shared by all CPU's
- 4) A single integrated operating system exercises overall control of hardware and software
(See Figure 7.4.3 and Figure 7-13).

7.4.1 SINGLE PROCESSOR

Figure 7-3 depicts a conventional single processor system. The major elements are the CPU, memory and Input-Output subsystems. The single processor has several shortcomings in the light of the overall node requirements, namely:

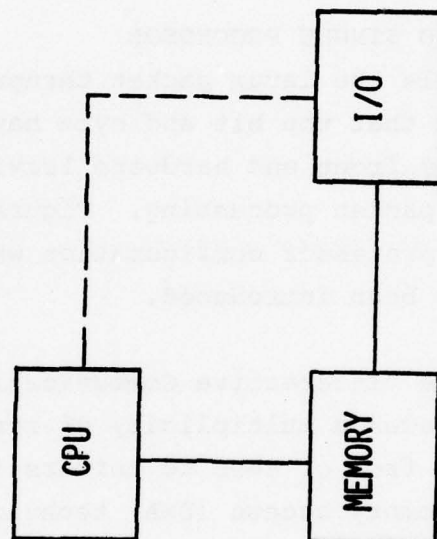


FIGURE 7.3 SINGLE PROCESSOR

- a) The overall thruput requirement for voice traffic for a large or super node is about 75 megabits. It is not reasonable to expect a single processor to cope with this requirement;
- b) A single processor is not very modular. One designed for a large node would be a relatively poor fit for a small node;
- c) In order to maintain communications in the event of failure, a backup processor is required. Although this is also true with multiple or multiprocessor systems the cost and size of the backup equipment is much greater in the case of the single processor.

7.4.1.1 MODIFIED SINGLE PROCESSOR

In order to handle the large packet thruput requirement, it becomes apparent that the bit and byte handling functions are best performed by front end hardware leaving the main processor free to perform packet processing. Figure 7-4 represents a modified single processor configuration wherein special front end hardware has been introduced.

RCA developed the "Interactive Communications Channel" (ICC). This unit interfaces a multiplicity of real time data links. Data is received from or sent to buffers in the processor memory via the direct memory access (DMA) technique. This unit which will be discussed in further detail para. 7.4.1.1.1 performs many of the ADCCP protocol functions and raises a processor interrupt only when a packet of data has been completely received or transmitted.

The ICC is now a working unit operating, in the ADPT testbed, as the interface link between access lines and trunks and main processor memory.

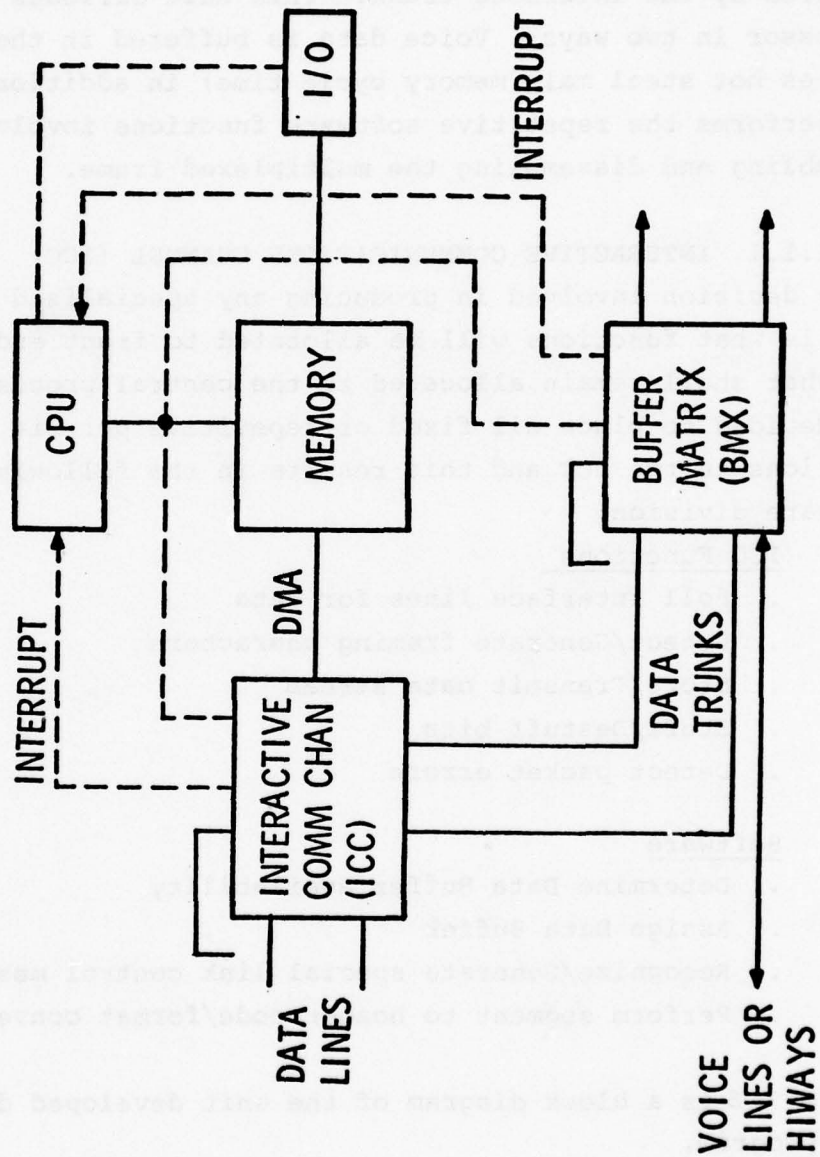


FIGURE 7-4 MODIFIED SINGLE PROCESSOR

RCA also developed, as part of this contract, a "buffer Matrix Unit" (BMX). This unit is a computer controlled dynamic multiplexor which buffers data and voice bits on individual channels and draws from these buffers the programmed sequence required by the internode trunk. This unit offloads the main processor in two ways: Voice data is buffered in the BMX and so does not steal main memory cycle time; in addition, the BMX performs the repetitive software functions involved in assembling and disassembling the multiplexed frame.

7.4.1.1.1 INTERACTIVE COMMUNICATIONS CHANNEL (ICC)

A key decision involved in producing any specialized front end unit is what functions will be allocated to front end hardware and what shall remain allocated to the central processor. It was decided to place all fixed or repetitive per bit or character functions in the ICC and this results in the following hardware/software division:

ICC Functions

- . Poll interface lines for data
- . Detect/Generate framing characters
- . Store/Transmit data stream
- . Stuff/Destuff bits
- . Detect packet errors

Software

- . Determine Data Buffer availability
- . Assign Data Buffer
- . Recognize/Generate special link control messages
- . Perform segment to header code/format conversions

Figure 7-5 is a block diagram of the unit developed during the ADPT program.

It is our judgement that the implemented ICC functions represent the minimum amount of functions that should be allocated to a front end processor. The ICC developed is a hardware device and

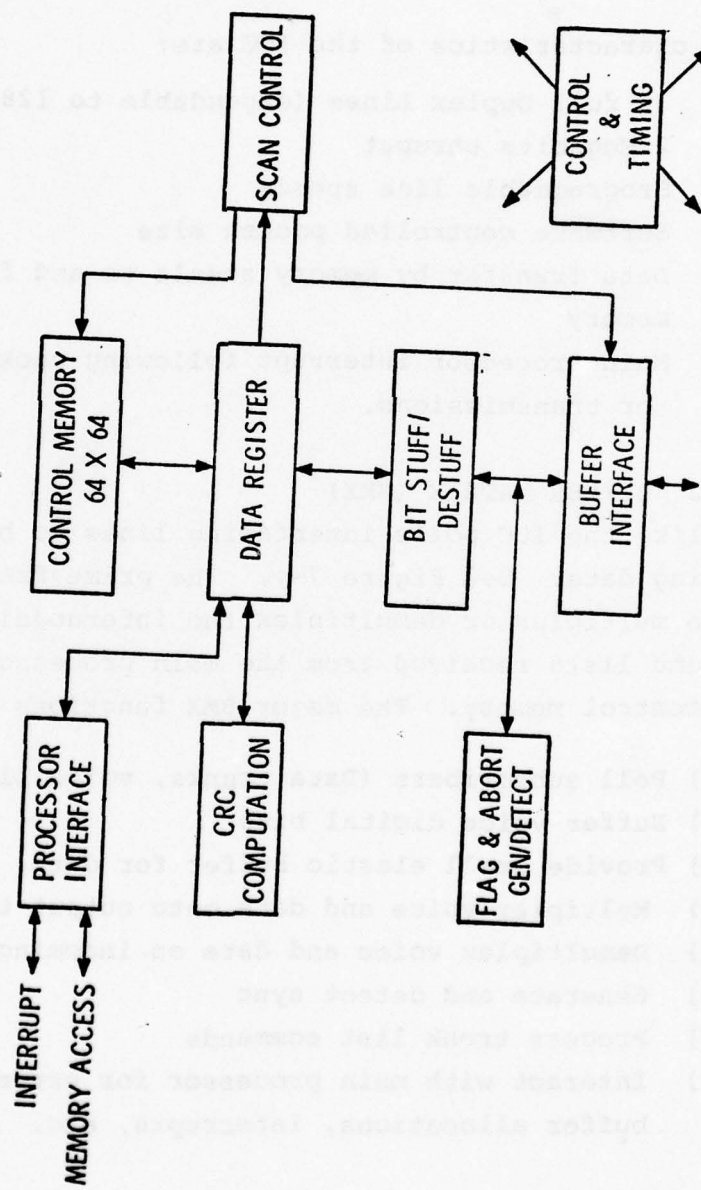


FIGURE 7-5 ICC BLOCK DIAGRAM

is not microprocessor controlled. For lower throughput systems a more complex front end device could include a microprocessor control and might pickup additional front end processing functions, however, the ICC is a high throughput device and provides at least a 15:1 reduction in processing load to the processor (by eliminating most bit and byte processing).

All candidate architectures will therefore include the ICC type device as part of the candidate configuration.

Specific characteristics of the ICC are:

- . 16 full Duplex Lines (expandable to 128)
- . 2 Megabits thruput
- . Programmable line speeds
- . Software controlled packet size
- . Data Transfer by memory steals to and from processor memory
- . Main Processor interrupt following packet reception or transmissions.

7.4.1.1.2 BUFFER MATRIX (BMX)

The BMX like the ICC polls interfacing lines to buffer incoming or outgoing data. See Figure 7-6. The prime function of the BMX is to multiplex or demultiplex the internodal trunk frame using frame lists received from the main processor and stored in the BMX control memory. The major BMX functions are as follows:

- 1) Poll subscribers (Data trunks, voice lines or highways)
- 2) Buffer voice digital bits
- 3) Provide small elastic buffer for data
- 4) Multiplex voice and data onto output trunks
- 5) Demultiplex voice and data on incoming trunks
- 6) Generate and detect sync
- 7) Process trunk list commands
- 8) Interact with main processor for error control, buffer allocations, interrupts, etc.

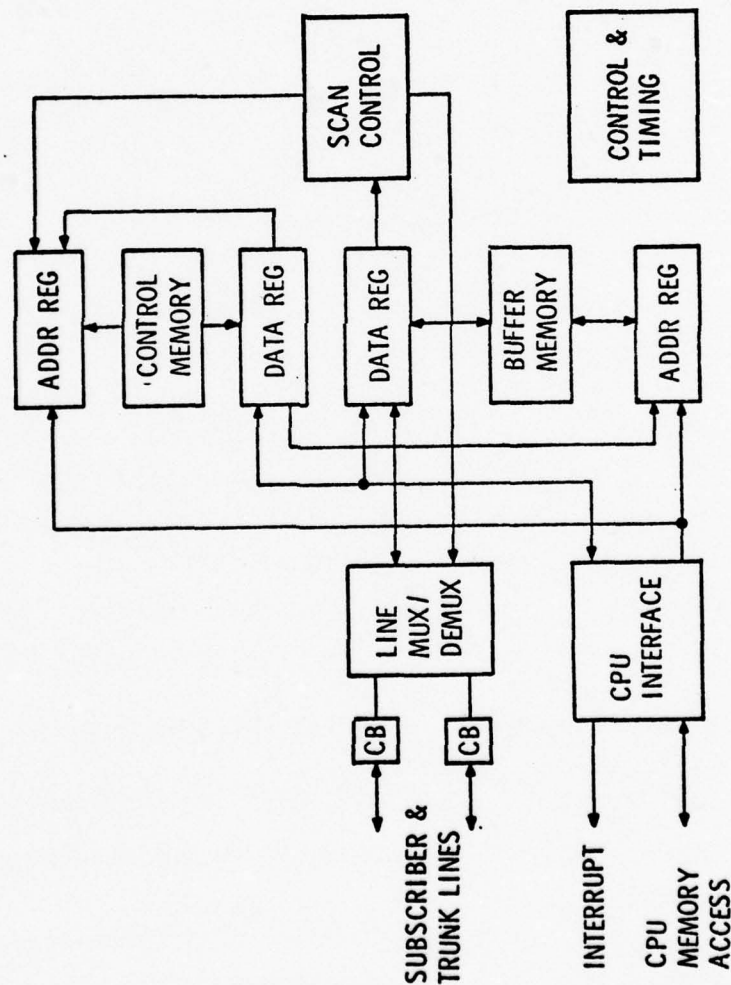


FIGURE 7-6 BMX BLOCK DIAGRAM

RCA has developed and tested a BMX unit for the ADPT testbed. This unit limits main processor handling of voice calls to . generation and detection of CCIS messages and control of trunk frame lists. The problems of real time voice data handling are removed from the main processor. For this reason the BMX type device like the ICC is an I/O concept that will be included in all future candidates under consideration, Detailed descriptions of the BMX and ICC are in Appendix I.

7.4.2 MULTIPLE PROCESSORS

7.4.2.1 MULTIPLE PROCESSORS WITHOUT SHARED MEMORY

Multiple Processors Systems deploy independent processors each performing fixed functions. Figure 7-7 depicts the basic multiple processor concept. The processors, in the basic multiple processor configuration, do not utilize shared memory. Interprocessor communication must be provided, figure 7-8 illustrates the various techniques that can be used. Table 7-4 illustrates the pros and cons of each approach. In general, the bussing approach is most flexible and very widely used, however, if the maximum number of processors in the configuration are few (i.e., four) direct I/O links also prove feasible, and if very long connection holding times are involved (i.e. minutes) switching techniques are useful.

Figure 7-9 illustrates the multiple processors candidate which appears to best qualify in its category.

Individual processors split the data load by dividing the number of lines and trunks under their direct control. The high speed bus is used to transfer data from buffers in the input processor to buffers in the output processor. Since input data will not use the bus and not all outputs require processor-processor transfer a bus that will provide a 3 MBS transfer rate should prove adequate, however, a 6 MBS bus is considered a safer design objective.

Based on the assumption that a processor can be designed to accommodate 400 packets/second a system containing 3-4 processors should prove adequate to handle the large nodes. Several stringent requirements would be placed on each individual processor, namely:

- a) The memory size capacity must be at least 256 kilobytes per processor;
- b) In order that each processor serve up to four T1 trunks at least four DMA ports would be required to serve

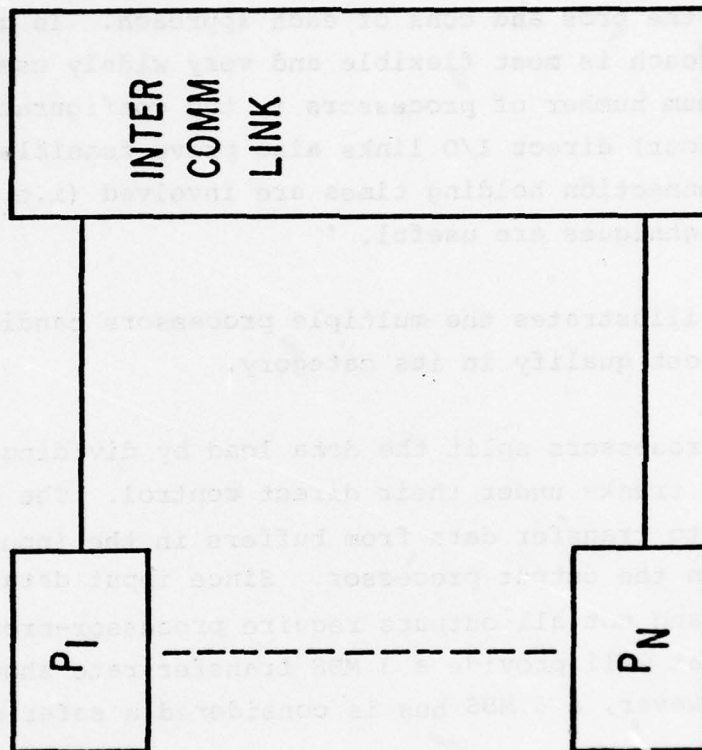
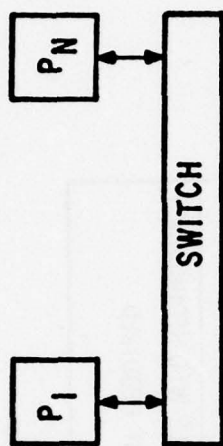
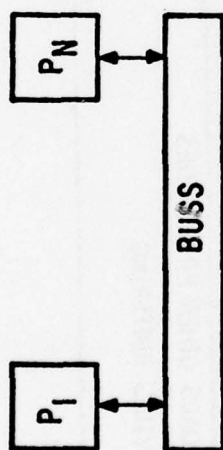


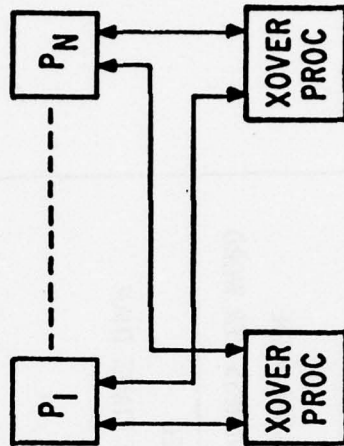
FIGURE 7-7 BASIC MULTIPLE PROCESSORS CONFIGURATION



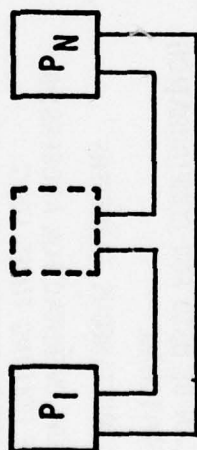
b)



a)



d)



c)

DIRECT I/O LINKS

FIGURE 7-8 INTERPROCESSOR COMMUNICATION LINKS

TABLE 7-4 INTERCOMM LINK TRADE MATRIX

	ADVANTAGES	DISADVANTAGES
BUSS	EXPANDABLE	<ul style="list-style-type: none"> • REQUIRES VERY HIGH SPEED CIRCUITS • COMMON FAILURE MODE • REQUIRES ADDRESS PER WORD
SWITCH	<ul style="list-style-type: none"> • CAN BE USED FOR CONFIGURATION SWITCHING • SIMULTANEOUS TRANSFERS • LOW OVERHEAD FOR ADDRESSING IF HOLDING TIMES LONG 	<ul style="list-style-type: none"> • SPEED LIMITED • LONGER WAIT TIMES THAN BUSSING
DIRECT I/O	OFTEN CHEAPEST IMPLEMENTATION	<ul style="list-style-type: none"> • EXPANDIBILITY LIMITED • DOES NOT ADAPT TO TRAFFIC FLOW REQUIREMENT
XOVER PROC	OFFLOADS OTHER PROCESSORS PROVIDES BUFFERING	<ul style="list-style-type: none"> • LEAST RELIABLE W/O REDUNDANCY • HIGH THRUPUT REQUIRED

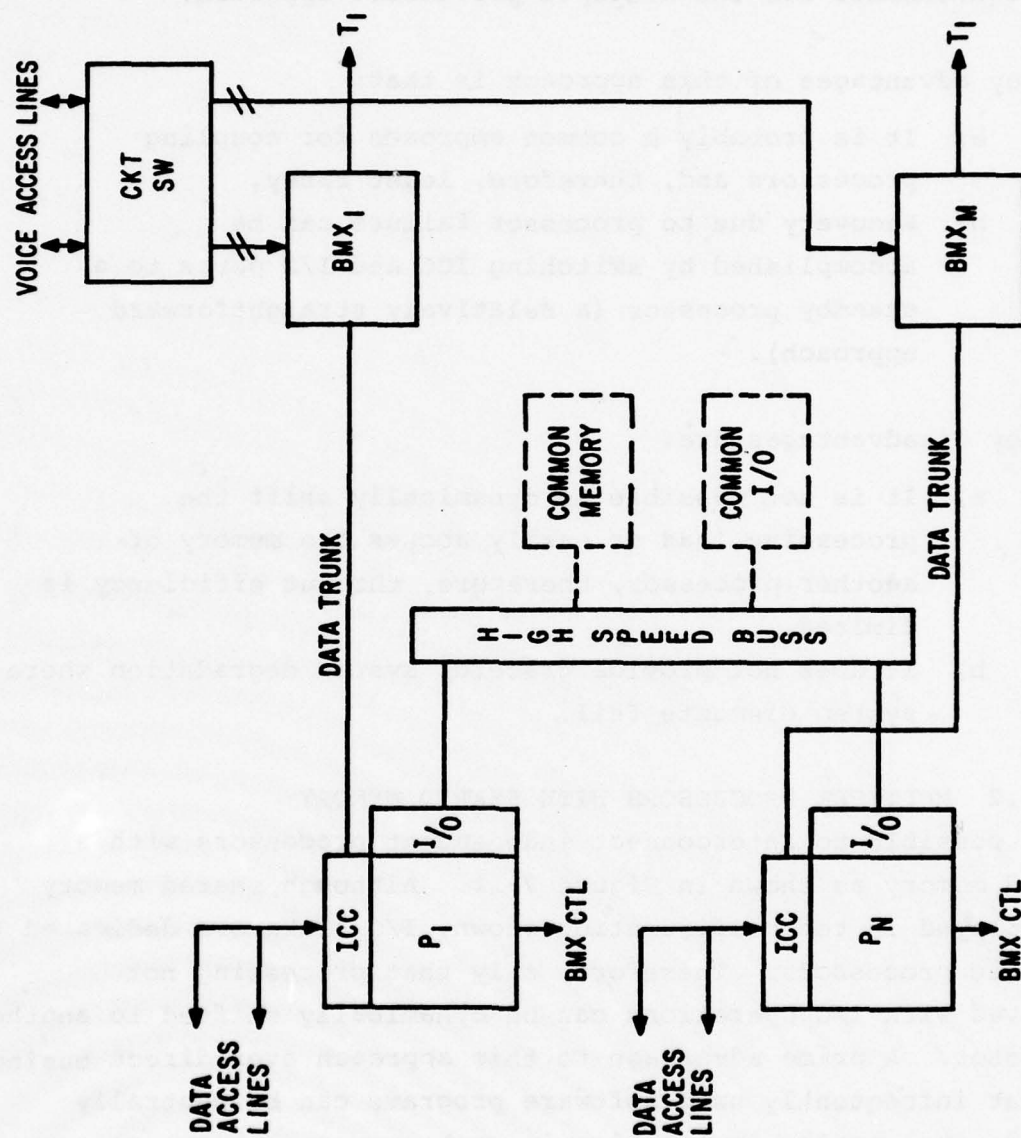


FIGURE 7-9 MULTIPLE PROCESSOR WITH HIGH SPEED BUSS

up to four ICC's per processor. Furthermore, the capability must be provided for the required 3-6 MBS interprocessor bus.

The DEC PDP-11 configuration illustrated in Figure 7-10 approaches the requirements for the multiple processors approach.

The key advantages of this approach is that;

- a) It is probably a common approach for coupling processors and, therefore, least risky,
- b) Recovery due to processor failure can be accomplished by switching ICC and I/O ports to a standby processor (a relatively straightforward approach).

The key disadvantages are:

- a) It is not possible to dynamically shift the processing load or easily access the memory of another processor, therefore, thruput efficiency is limited.
- b) It does not provide graceful system degradation where system elements fail.

7.4.2.2 MULTIPLE PROCESSORS WITH SHARED MEMORY

It is possible to interconnect independent processors with a shared memory as shown in Figure 7-11. Although shared memory is employed in the configuration shown, I/O links are dedicated to specific processors. Therefore, only that processing not involved with I/O operations can be dynamically shifted to another processor. A prime advantage to this approach over direct busing is that infrequently used software programs can be centrally stored, thus saving duplication in each processor memory.

The key disadvantage is that extra memory transfer is required to move data between processors, and the shared memory is more complex from the physical hardware standpoint than a bus.

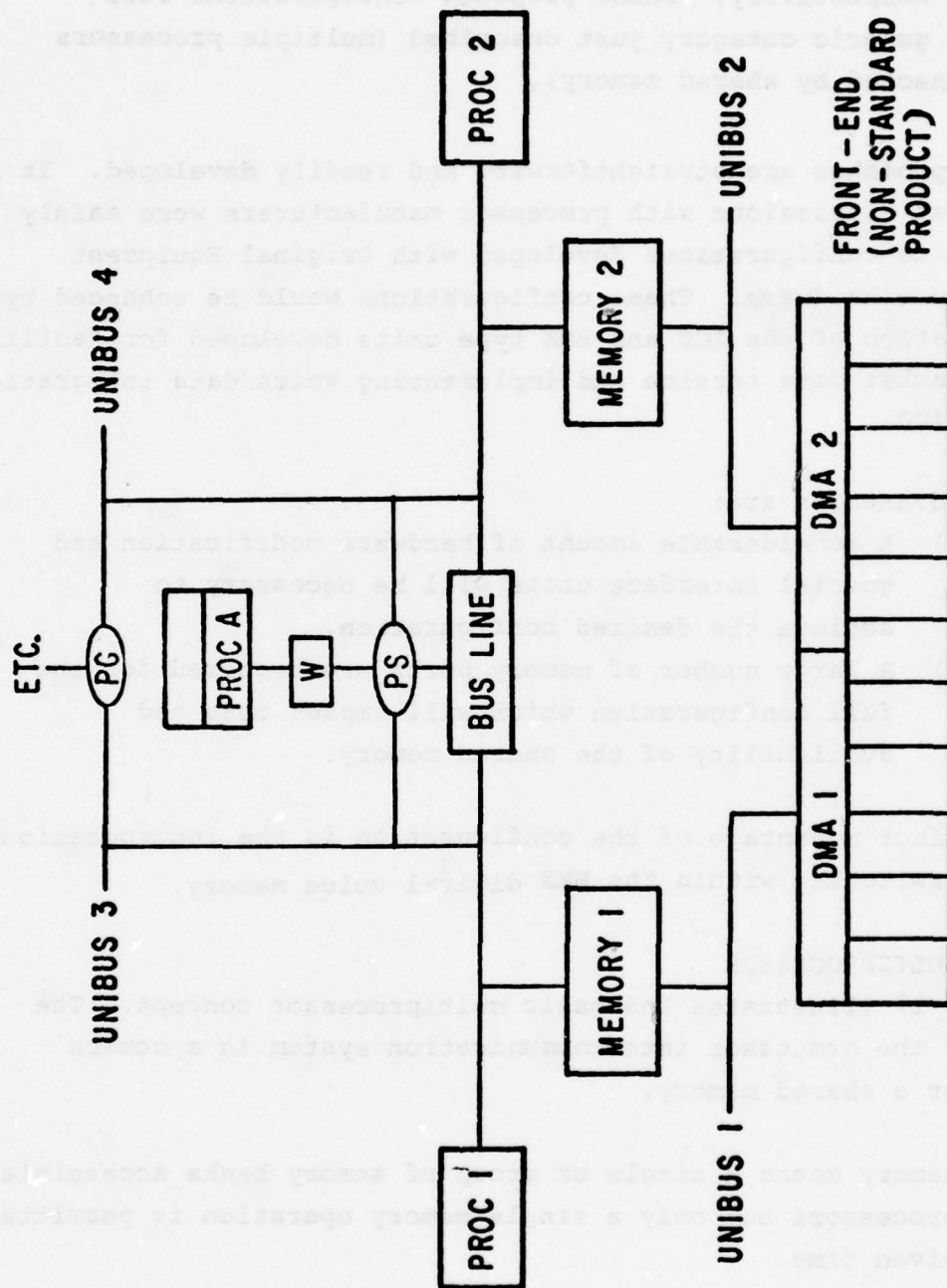


FIGURE 7-10 DEC PDP-11 CONFIGURATION

Discussion between RCA and representatives of Varian and Data General led to suggested configurations shown in Figures 7-11 and 7-12 respectively. These proposed configurations fall into the generic category just described (multiple processors interconnected by shared memory).

These approaches are straightforward and readily developed. It is noted that discussions with processor manufacturers were mainly confined to configurations developed with Original Equipment Manufacture hardware. These configurations would be enhanced by incorporation of the ICC and BMX type units developed for facilitating packet data service and implementing voice/data integration. integration.

The disadvantages are:

- 1) A considerable amount of hardware modification and special interface units will be necessary to achieve the desired configuration,
- 2) A large number of memory ports are required for the full configuration which will impact cost and availability of the shared memory.

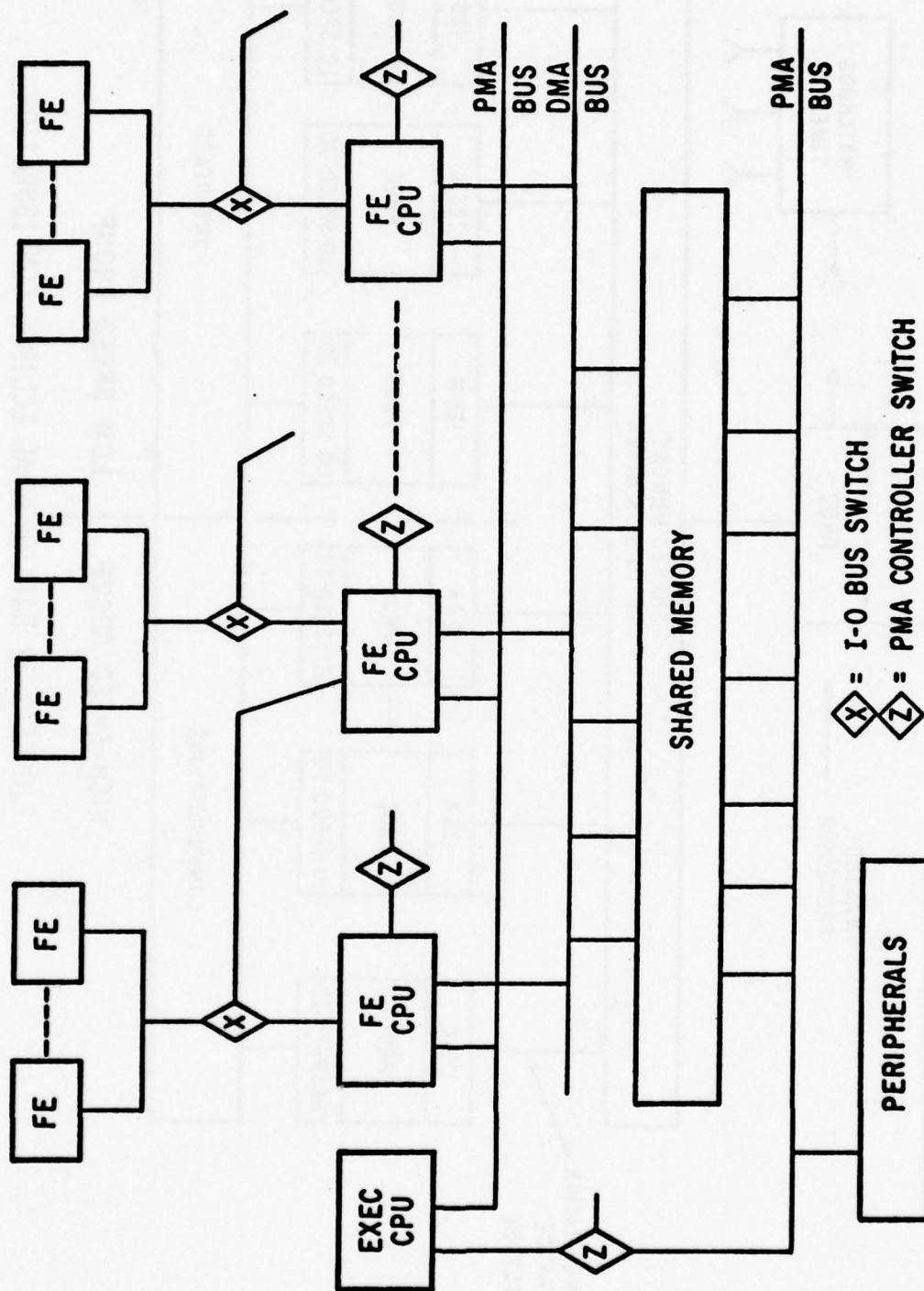
One distinct advantage of the configuration is the incorporation of circuit switching within the BMX digital voice memory.

7.4.3 MULTIPROCESSOR

Figure 7-13 illustrates the basic multiprocessor concept. The heart of the processor intercommunication system is a common memory or a shared memory.

Common memory means a single or group of memory banks accessible to all processors but only a single memory operation is permitted at any given time.

Shared memory means a multiplicity of memory banks each accessible to all processors but also capable of interleaved operation, that



FE-CPU = FRONT END CPU

FIGURE 7-11 VARIAN CONFIGURATION

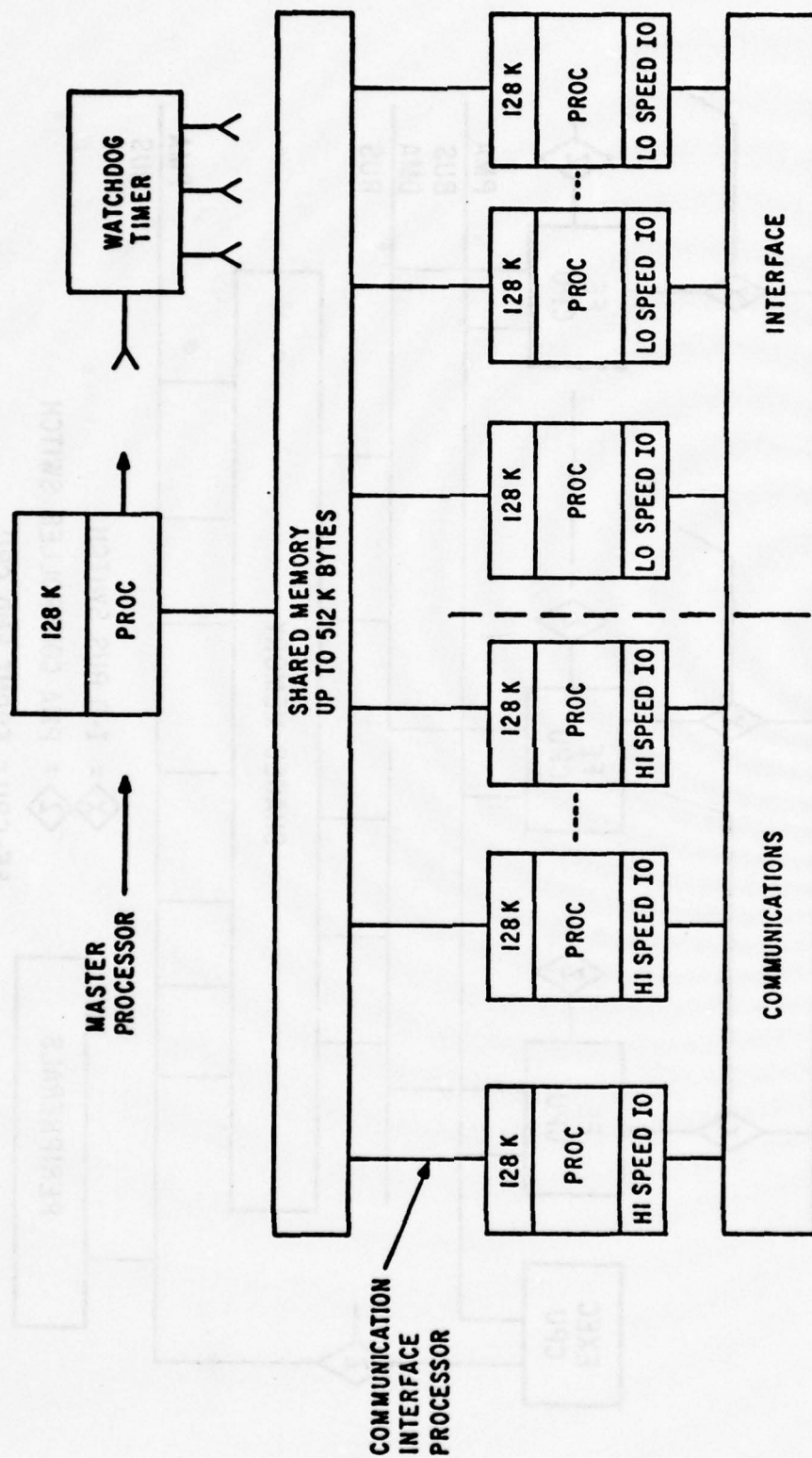


FIGURE 7-12 DATA GENERAL ECLIPSE CONFIGURATION

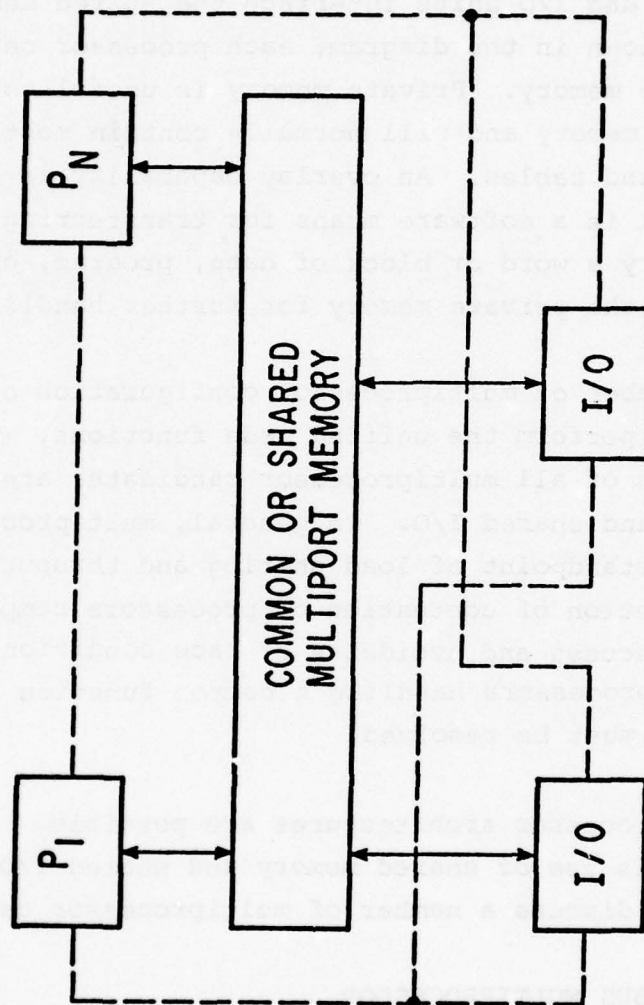


FIGURE 7-13 BASIC MULTIPROCESSOR

is, each memory bank can simultaneously perform memory read/write operations. Shared memory will improve thruput over common memory but the cost is slightly higher. In this system where thruput performance is so important, the use of shared memory approach is necessary to increase thruput.

All processors and I/O units interface the shared memory. Although not shown in the diagram, each processor can and should contain private memory. Private memory is useful to reduce conflict for main memory and will normally contain most frequently used programs and tables. An overlay capability is also essential; that is a software means for transferring from shared or common memory a word or block of data, program, or table information to the private memory for further handling.

There are a number of multiprocessor configuration candidates that can be used to perform the unified node functions. The key common features of all multiprocessor candidates are the use of shared memory and shared I/O. In general, multiprocessors are good from the standpoint of load sharing and thruput efficiency, however, resolution of contention of processors competing for memory or I/O access and avoidance of race conditions resulting from separate processors handling a common function are special problems which must be resolved.

Various multiprocessor architectures are possible. What is common to all is use of shared memory and shared I/O. The following paragraphs discuss a number of multiprocessor candidates.

7.4.3.1 MULTIBUS MULTIPROCESSOR

Figure 7-14 illustrates a multibus multiprocessor similar to the ARPA "pluribus" architecture. (See Figure 7-15) Individual processors communicate with separate memory busses and separate I/O busses via bus couplers. In Figure 7-14, the special I/O developed by RCA for handling packet switching and dynamic channel

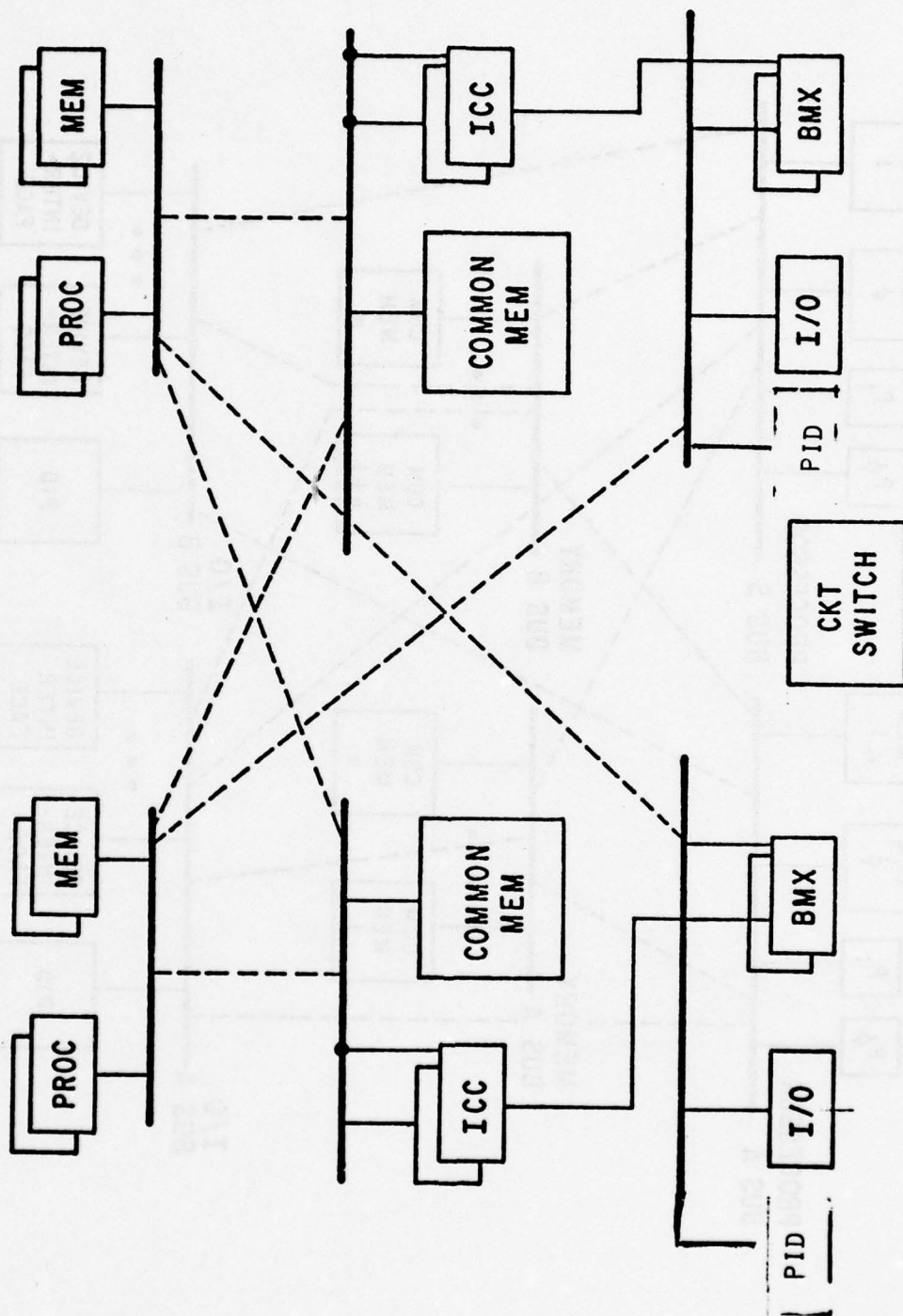


FIGURE 7-14 MULTIBUS MULTIPROCESSOR

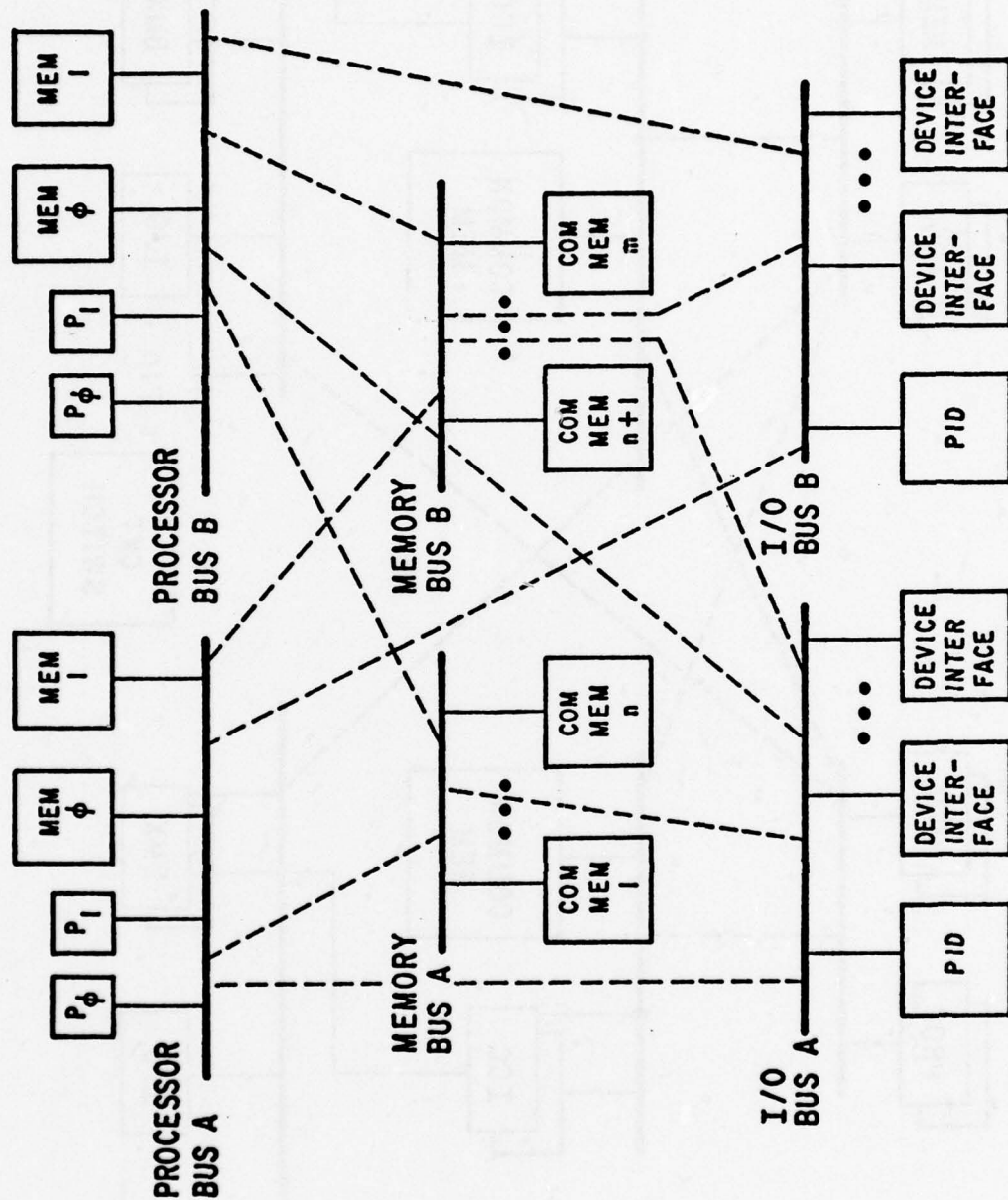


FIGURE 7-15 PLURIBUS SYSTEM CONFIGURATION

allocation on trunks; namely the ICC and BMX units have been incorporated into the architecture. Since the ICC communicates via DMA only with memory, the ICC is tied to the memory bus rather than the I/O buss. The location of the BMX unit will depend on the final BMX design, but can probably be handled as an ordinary peripheral device communicating directly with processors dedicated to the circuit switch function.

Some form of interrupt hardware is required to assign tasks to processors. A pseudo interrupt device (PID) is one such mechanization. Other mechanizations are possible and will be discussed in the discussion of interrupt structure in the software architecture section of this report.

The multiprocessor system is very versatile. Any processor can perform any task. Theoretically, any single failure in the system can be bypassed, and can be coupled from the rest of the system.

The key problems are:

- a) The buss structure hardware, which includes a multiplicity of couplers and arbitrators, is extremely complex,
- b) The large number of busses employed make the analyses of system thruput more complex. A great amount of engineering and software design effort will be needed to assure minimization of contention for any given buss.
- c) Due to the system organization, several overhead processing is incurred in linking small task segments, therefore, system thruput will be relatively lower.
- d) In addition to higher development cost, software maintenance is also more complex and should result in higher cost.

7.4.3.2 MULTIPROCESSOR WITH MULTIPORT MEMORY AND CENTRALIZED BMX

Figure 7-16 illustrates a multiprocessor approach configured with shared memory as the main intercommunications means. Shared memory permits multiple users to access the memory and is divided into banks wherein independent memory banks can be simultaneously accessed, thus improving system throughput. One of the chief advantages of using a shared memory is that a number of memory manufacturers market these units. This eliminates the need for buss couplers designs.

As shown in the figure, the data and voice memories are made independent. This appears to be desirable. Since most processing involves data access, separation of data and voice memories reduces processor memory access conflicts.

The approach is relatively straightforward. Key problems with this configuration are:

- a) Since each memory bank must contain port arbitration logic, unit costs rise proportionately with the increase in the number of ports. Presently offered by off-the-shelf CPU systems offer only limited number of ports, usually ranging from 2-8;
- b) The central BMX approach will exhibit single point failures unless redundant BMX hardware and switch-over logic is employed. An alternate approach is to employ smaller independent BMX modules.

A key advantage of this configuration (other than its inherent flexibility) is that data is written into memory only once, and no intermediate data transfers between memory modules are necessary.

7.4.3.3.3 MULTIPROCESSOR WITH MULTIPORT MEMORY AND DECENTRALIZED BMX

In order to read or write an eight bit character to or from the BMX, the BMX is involved with transfer decisions and address lookups. On the average, about 5 memory accesses are required per

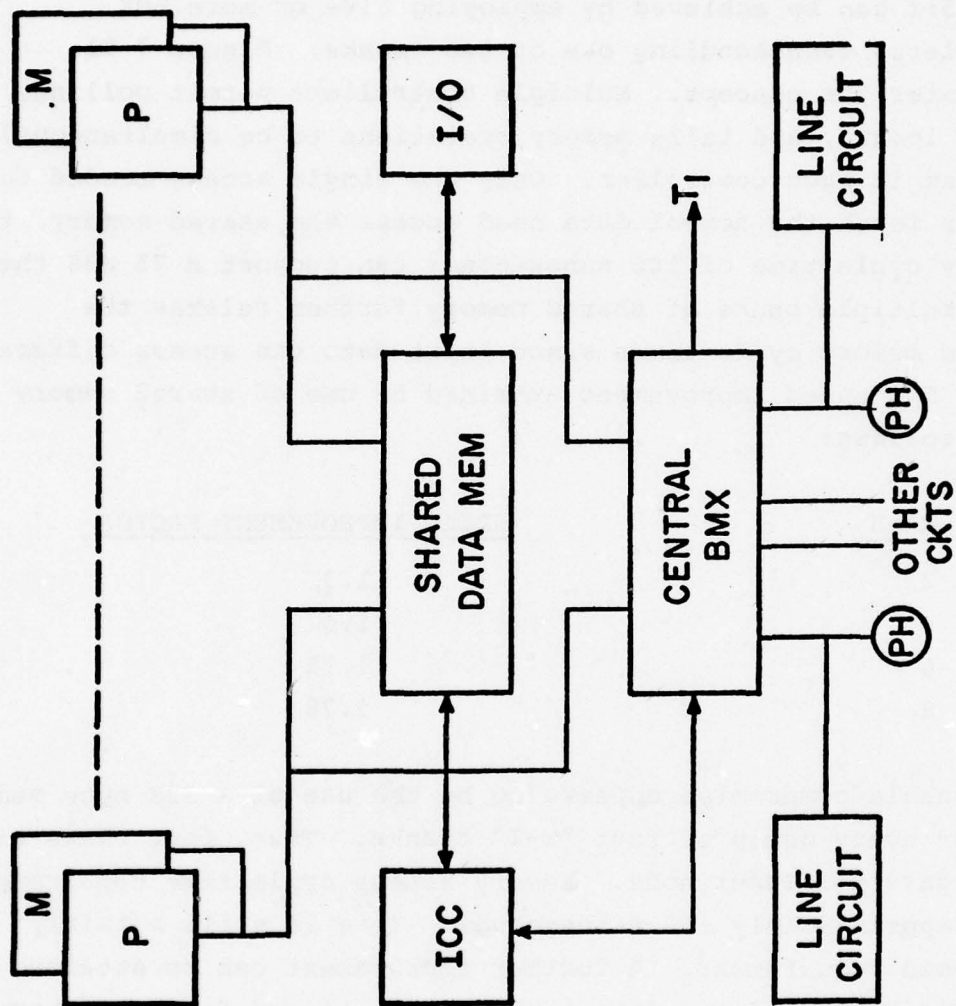


FIGURE 7-16 MULTIPROCESSOR WITH MULTIPOINT MEMORY AND
CENTRAL BMX

octet transfer. In order to service a super node with a thruput in the order of 75 MBS, approximately 46 million accesses per second or 21 nanosecond memory access time is required. This is a difficult and expensive requirement to meet. An improvement of nearly 5:1 can be achieved by employing five or more BMX controllers, each handling one or two trunks. Figure 7-21 illustrates the concept. Multiple controllers permit polling, address lookup, and tally memory operations to be simultaneously performed in each controller. Only the single access needed to store or fetch the actual data need access the shared memory, thus a memory cycle time of 100 nanoseconds can support a 75 MBS thruput. Use of multiple banks of shared memory further relaxes the required memory cycle times since input data can access different banks. The speed improvement attained by use of shared memory are as follows:

<u># BANK</u>	<u>SPEED IMPROVEMENT FACTOR</u>
2	1.3
4	1.6
6	1.75
8	1.78

A reasonable compromise appears to be the use of a 32K byte memory bank for every group of four "T-1" trunks. Thus, four banks will accommodate the super node. Memory access cycle time requirement is now approximately 160 nanoseconds. This is still a fairly high speed requirement. A further improvement can be attained by employing half word or sixteen bit access to and from the shared memory. Normally a 2:1 speed gain over byte access is attained, however, if the trunk slot size happens to be an odd number of bytes, the BMX controller must access the last byte separately. Assuming that the trunk frame period is 10 milliseconds, the only potential voice rate that will require this odd byte access is 2400 Baud, and then it will only occur 50% of the time. Considering this fact and also recognizing that if more than 25% of the voice traffic at a node is 2400 baud, a substantial reduction in

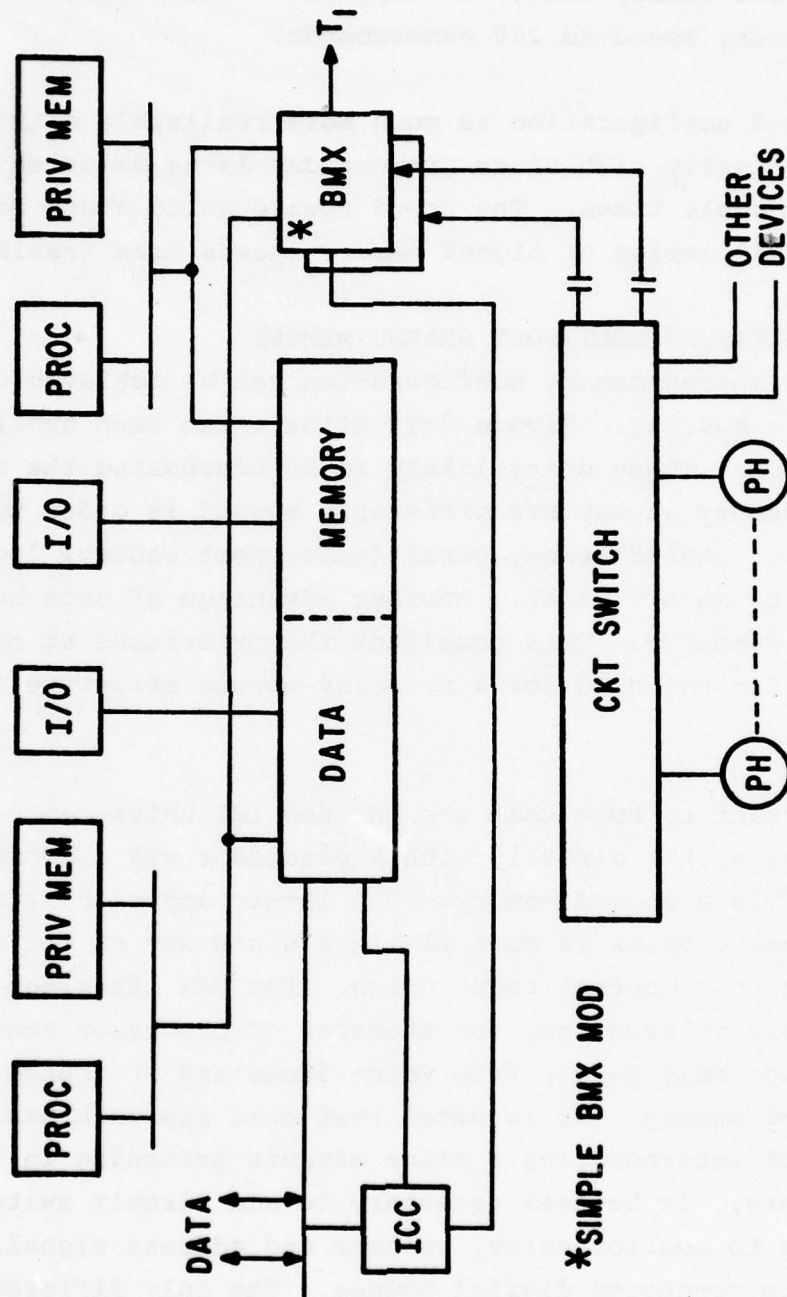


FIGURE 7-17 MULTIPROCESSOR WITH MULTIPOINT MEMORY AND
DECENTRALIZED BMX

projected total bandwidth (hence memory speed) is required, it is safe to assume an 80% relaxation in memory speed requirement if 16 bit wide memory access is employed. This raises the required memory speed to 288 nanoseconds.

This improved configuration is much more realizable although memory cost may be fairly high since present day large memories normally have slower cycle times. The trend toward solid state memory makes the realization of higher memory speeds more feasible.

7.4.3.4 MULTIBUSS MULTIPORT SHARED MEMORY

An improved shared memory configuration can be achieved by employing intermediate bussing. Figure 7-18 illustrates such hybrid configuration. Those units likely to be addressing the same bank of shared memory anyway are preferably bussed in order to reduce the number of shared memory ports (since port control logic is duplicated by memory bank). Another advantage of such bussing is achieving symmetry. This equalizes the priorities of each memory port obviating the need for a priority access structure for memory access.

It is important to note that the BMX and ICC units can be designed to interface either directly with a processor via a common bus or indirectly via a shared memory. The latter approach although slightly more complex is more flexible since any of the system processor(s) can control these units. The BMX interface to data shared memory is used only for transfer of processor control data. Digital Voice only passes from voice lines and or trunks to the Voice shared memory. It is noted that this approach has the advantage of incorporating a voice circuit switching in the shared memory. It becomes necessary to add circuit switch (C/S) controllers to monitor seize, release and address signalling derived from connected digital phones. The only difference between this approach and a typical processor controlled space division circuit switch is that data switching within the Voice shared memory replaces circuit switch crosspoints.

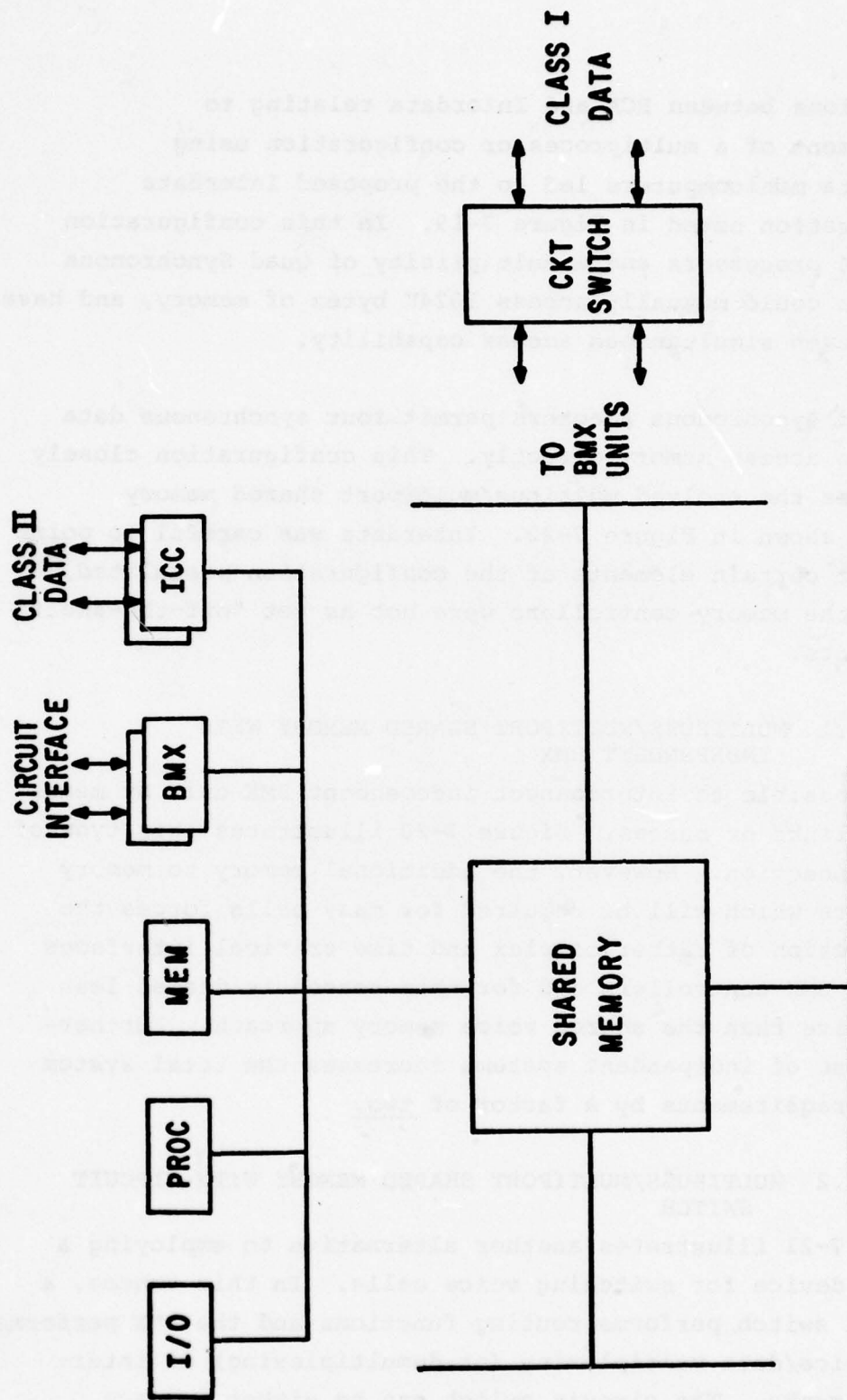


FIGURE 7-18 MULTIBUSS/MULTIPOINT SHARED MEMORY

Discussions between RCA and Interdata relating to development of a multiprocessor configuration using Interdata minicomputers led to the proposed Interdata configuration noted in Figure 7-19. In this configuration up to 14 processors and a multiplicity of Quad Synchronous Adaptors could mutually access 1024K bytes of memory, and have up to seven simultaneous access capability.

The Quad Synchronous Adapters permit four synchronous data links to access memory directly. This configuration closely resembles the evolved multibus/multiport shared memory concept shown in Figure 7-22. Interdata was careful to point out that certain elements of the configuration postulated, namely the memory controllers were not as yet "off-the-shelf" components.

7.4.3.4.1 MULTIBUSS/MULTIPOINT SHARED MEMORY WITH INDEPENDENT BMX

It is possible to interconnect independent BMX unit by means of I/O links or busses. Figure 7-20 illustrates this type of interconnection. However, the additional memory to memory transfers which will be required for many calls forces the introduction of rather complex and time critical interfaces between BMX controllers and for this reason is deemed less attractive than the shared voice memory approach. Furthermore, use of independent systems increases the total system buffer requirements by a factor of two.

7.4.3.4.2 MULTIBUSS/MULTIPOINT SHARED MEMORY WITH CIRCUIT SWITCH

Figure 7-21 illustrates another alternative to employing a memory device for switching voice calls. In this scheme, a circuit switch performs routing functions and the BMX performs only voice/data multiplexing (or demultiplexing) on inter-nodal trunks. The circuit switch can be either a space

division or a time division switch. Since the voice data rates are not defined and could include a mix of various rates, the space division approach will be given a prime consideration. (However, if further definition of traffic inputs indicates a predominant voice rate, those calls operating at the most common rate could be switched employing a time division switch.)

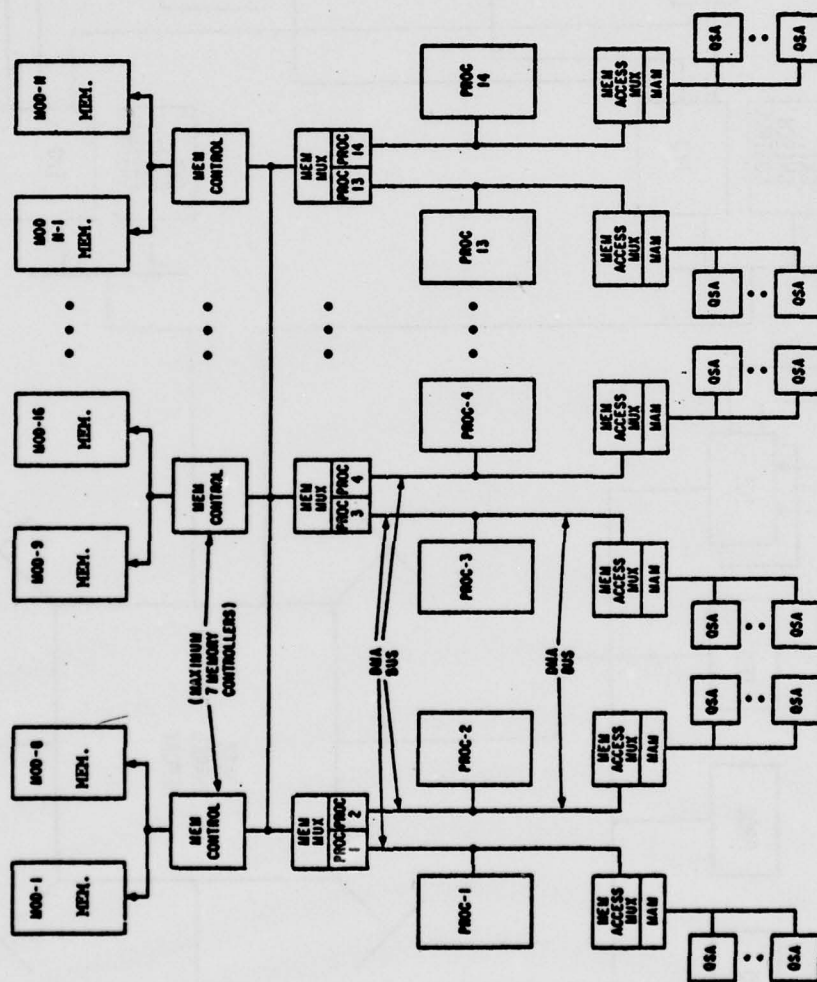


FIGURE 7-19 INTERDATA MULTIPROCESSOR CONFIGURATION

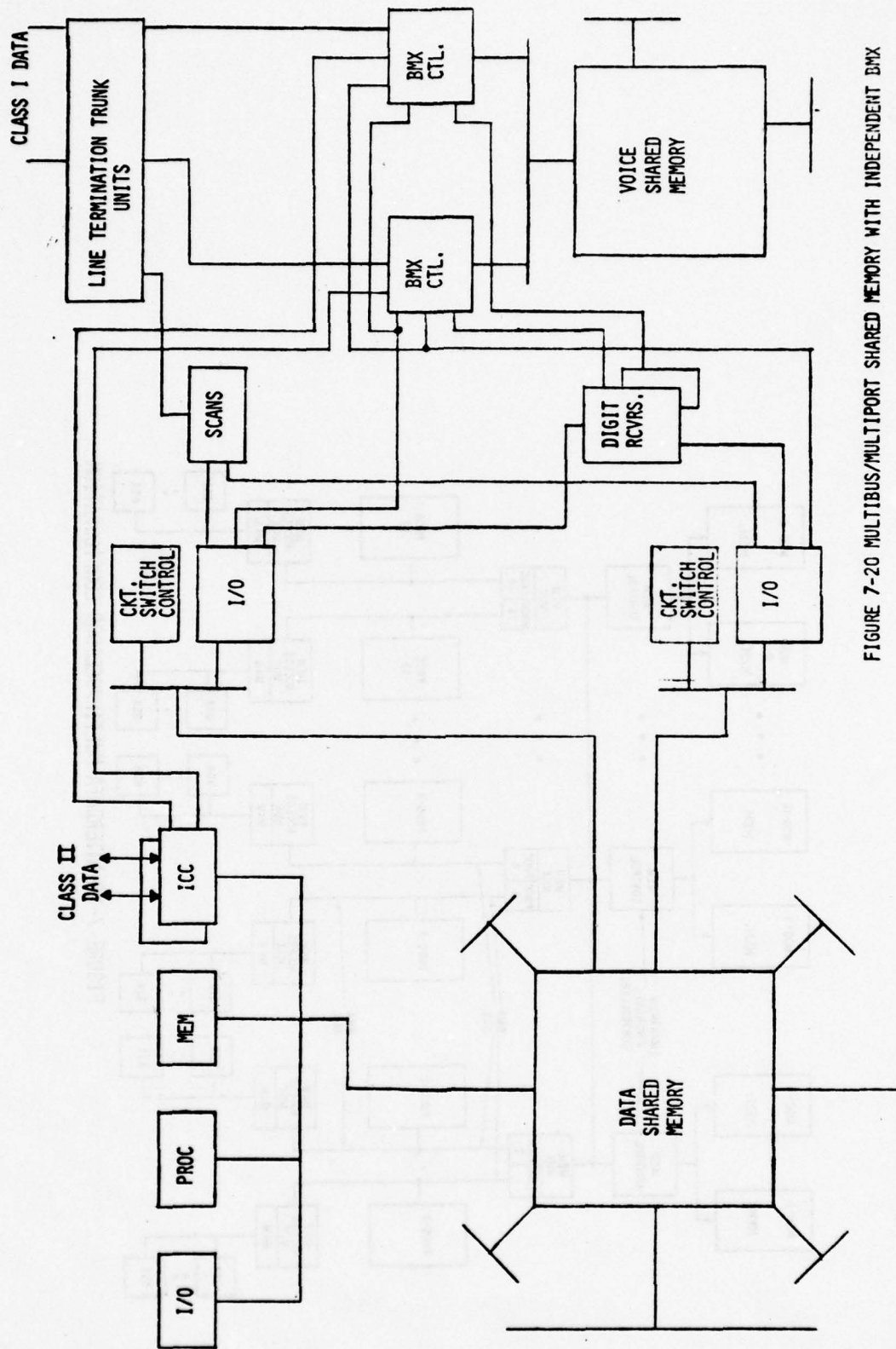


FIGURE 7-20 MULTIBUS/MULTIPOINT SHARED MEMORY WITH INDEPENDENT BMX

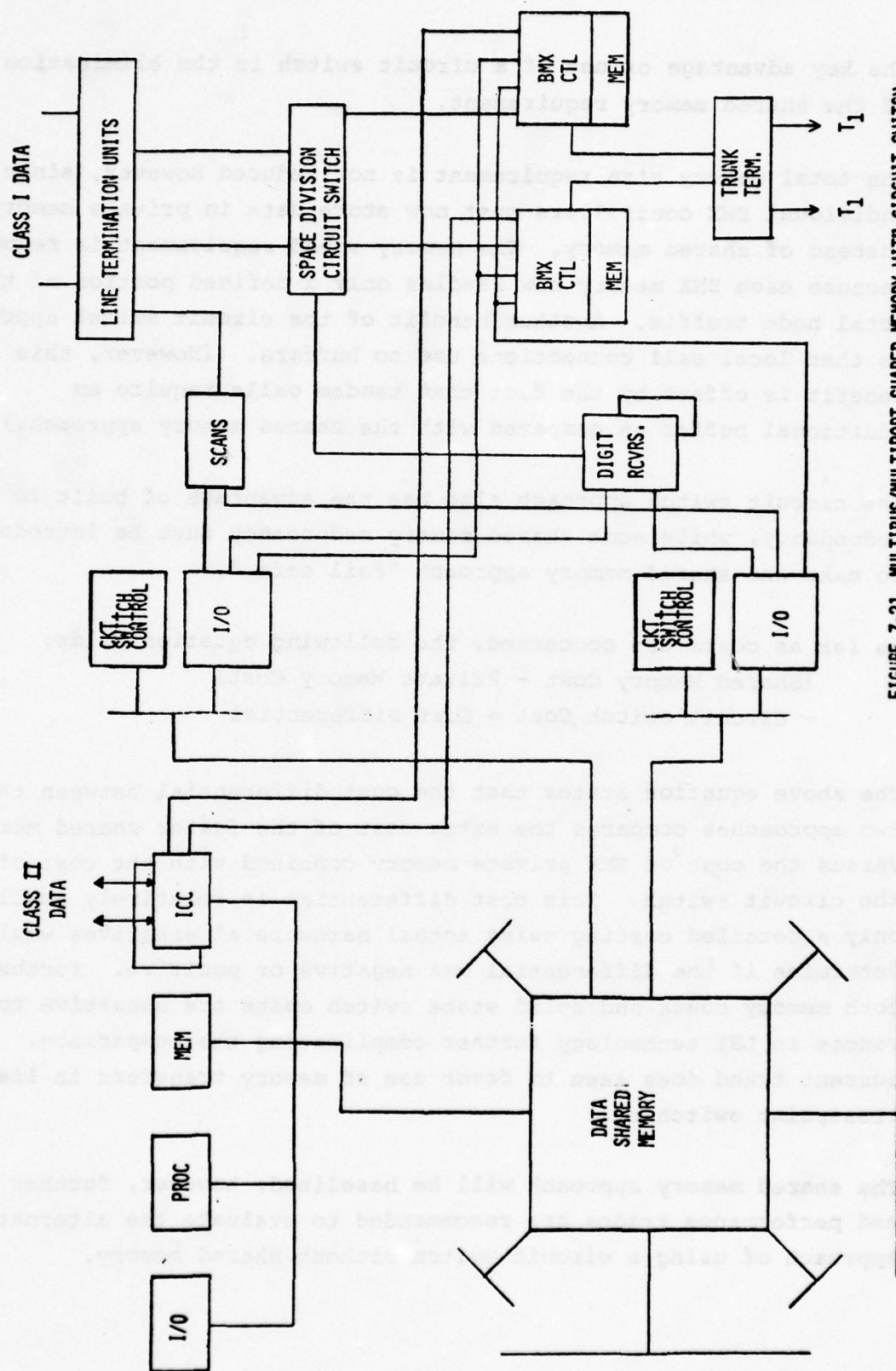


FIGURE 7-21 MULTIBUS/MULTIPORT SHARED MEMORY WITH CIRCUIT SWITCH

The key advantage of use of a circuit switch is the elimination of the shared memory requirement.

The total memory size requirement is not reduced however, since individual BMX controllers must now store data in private memory instead of shared memory. The memory speed requirement is reduced because each BMX memory now handles only a defined portion of the total node traffic. Another benefit of the circuit switch approach is that local call connections use no buffers. (However, this benefit is offset by the fact that tandem calls require an additional buffer as compared with the shared memory approach.)

The circuit switch approach also has the advantage of built in redundancy, while some shared memory redundancy must be introduced to make the shared memory approach "fail safe."

As far as costs are concerned, the following equation holds:

$$\begin{aligned} & (\text{Shared Memory Cost} - \text{Private Memory Cost}) \\ & - \text{Circuit Switch Cost} = \text{Cost Differential} \end{aligned}$$

The above equation states that the cost differential between the two approaches compares the extra cost of the faster shared memory versus the cost of RMX private memory combined with the cost of the circuit switch. This cost differential is relatively small and only a detailed costing using actual hardware alternatives would determine if the differential was negative or positive. Furthermore, both memory costs and solid state switch costs are sensitive to advances in LSI technology further complicating the comparison. The current trend does seem to favor use of memory transfers in lieu of crosspoint switching.

The shared memory approach will be baselined; however, further cost and performance trades are recommended to evaluate the alternate approach of using a circuit switch without shared memory.

7.5 EVALUATION

The method of evaluation of the various candidate architectures involves a process of establishing evaluation criteria, weighting the importance of each criterion for this evaluation, and then grading each candidate against each criterion. Finally a weighted sum figure of merit was developed for each candidate configuration. It is important to note that a small group of qualified personnel independently, then jointly participated in the ranking of criteria and grading of the candidates.

7.5.1 EVALUATION CRITERIA

The following criteria were chosen for the evaluation of candidates:

(1) Thruput or Transfer Rate

The total number of information bits transferred by the node per unit of time. For this evaluation, we are interested in the processing power of each system organization in terms of the thruput it can handle.

(2) Availability

The portion of a selected time interval during which the node is capable of performing its function. For this evaluation, we are interested in the ability to maintain service or partial service after an equipment failure.

(3) Reliability

The probability that the equipment will perform without failure for a specified time period.

(4) Expandability

The ability to enlarge or increase the performance capability of a node sometime in the future.

(5) Modularity

A characteristic of the nodal architecture whereby groups of common units or modules are incorporated which can be easily added or subtracted in order to achieve performance capability commensurate with a wide range of node sizes.

(6) Flexibility

The ease with which the system can be changed and improved in the future through software changes.

(7) Maintainability

The rapidity with which faults can be detected, isolated and corrected.

(8) Cost

The cost to acquire, install, operate and maintain the nodal hardware and software (i.e., life cycle cost).

(9) Technical Confidence (Feasibility)

The probability that the hardware and software can be implemented without major cost and schedule overruns.

7.5.2 CRITERIA RANKING

The most important factors are considered to be cost, technical risk, thruput (performance), and modularity. These were equally ranked and given a weight of "3".

The next group in importance are availability, maintainability, flexibility and expandability and were given a weight of "2".

Reliability is very important, however, all candidates will utilize modern solid state technology, therefore, reliability is considered of lesser importance in comparing the candidates and is given a weight of "1".

7.5.3 CANDIDATE EVALUATION

Table 7-5 presents a summary evaluation matrix wherein each candidate was graded to the degree it satisfied a particular criterion. Grades are figures of merit, the higher the number the better the candidate satisfies the objective. Thus, for example, a candidate whose projected cost is relatively low would be graded higher against the "cost" criterion.

It is important to note that the greatest degree of uncertainty lies in the estimation of relative costs. Candidates which should

TABLE 7- 5 ARCHITECTURE TRADE MATRIX

WTS	SP	MP	MB	SM/CBMX	SN/DBMX	H/DBMX
3 MODULARITY	1	4	3	2	4	4
2 EXPANDIBILITY	1	3	2	2	3	4
3 RISK	1	4	2	1	2	3
3 THRUPUT	0	2	3	0	4	4
1 RELIABILITY	4	3	2	4	3	3
2 AVAILABILITY	1	2	4	3	4	4
2 MAINTAINABILITY	3	4	2	3	3	3
3 COST	4+	4	2	4	2	3
2 FLEXIBILITY	2	2	4	3	4	4
SUM	17	28	24	25	29	32
WEIGHTED SUM	36	59	56	46	67	75

0 TOO RISKY
1 POOR
2 ADEQUATE
3 GOOD
4 BEST

SP - SINGLE PROCESSOR
MP - MULTIPLE PROCESSOR
MB - MULTIBUS
SM - SHARED MEMORY
H - HYBRID
CBMX - CENTRAL BMX
DBMX - DECENTRALIZED BMX

be theoretically lower in cost than others may prove to be higher when detailed cost analysis is performed. The reason this may occur is that different processor system manufacturers have developed processing systems for other applications and may already have developed an acceptable configuration which will satisfy the basic requirements.

The less new development is required the lower the selling price. Thus, it is to the benefit of the government to permit each offeror to bid his architecture. Relative cost estimates for this evaluation are based primarily on engineering judgement as to the system hardware and software complexity. Since cost is a very significant factor, those architectures which can satisfy the system requirements and rank close in evaluation to the recommended architecture remain viable candidates.

The evaluation does point up certain critical deficiencies in some candidates and permits a number of conclusions:

- (1) The full configuration should contain between four to eight processors.
 - (2) It does appear desirable to perform voice circuit switching employing a shared memory dedicated to voice which is accessed by a multiplicity of BMX controllers.
 - (3) The multiprocessor configuration appears to offer best performance as regards thruput and flexibility.
 - (4) The most efficient and balanced multiprocessor arrangement is a hybrid employing bussing and shared memory.
 - (5) Multiple processor configurations can meet data node performance requirements and are in wider current usage than the multiprocessor arrangements.
- However, the multiprocessor arrangement is a better configuration from the standpoint of potential thruput performance and system flexibility. The

multiprocessor approach is the recommended system provided cost differentials do not turn out to be inordinately high, in which the case multiple processor architectures are acceptable for the data network.

7.6 RECOMMENDED ARCHITECTURE

7.6.1 PROCESSING REQUIREMENTS

Although most of the candidate architectures, except the single processor, are acceptable the best candidate based on our evaluation is the true multiprocessor employing shared memory and judicious use of bussing. Figure 7-20 illustrates the recommended approach.

All real time data entering the node passes directly to the shared memory. Any of the processors can take over processing of the input packet or command. A modular distributed processor system is recommended which can be expanded from one to eight processors. A single processor system can be designed to handle at least 200 packets per second. Eight processors, if configured with private memory should be capable of slightly less than 8X the thruput of one processor or nearly 1600 packets/second. At least two independent banks of shared memory will be required so as to maintain operation in the event of shared memory failure. The smallest system will require at least two processors for the same reason.

Since the processing of CCIS packets for voice call handling represents a very small percentage of the total processing load and since it is important to provide a focal point for control of the recommended distributed BMX system, it is further recommended that two processors be dedicated to the handling of CCIS packets. Either processor will be able to control all voice switching.

Either by communicating directly to distributed BMX controllers or indirectly via voice shared memory, those processors handling CCIS traffic will control the frame format on the internodal trunks.

7.6.2 BMX REQUIREMENTS

Based on present day memory speeds and the need to develop a modular system, it is suggested that each BMX controller handle no greater

thruput capacity than 6.5 megabits per second. This means that the BMX controller can handle up to one full duplex T-1 trunk group and about 100 full duplex 16 KBS access lines or turnks. The BMX design will not be limited to T-1 or 16 KBS rates but should be able to handle any rate mix totaling less than 6.5 MBS thruput. A maximum port capacity of 128 appears to be a resonable constraint on BMX ports (that is to say, each BMX can interface up to 128 full duplex ports).

If the maximum system employs 16 T-1 trunks, 16 BMX controllers are required. If it is assumed that about a third of the voice traffic is tandem trunk traffic the maximum node can support 1140 erlangs of voice traffic (maximum) or erlangs average (1% Blocking). This is an adequate maximum capacity. Should any locale require a larger node, several nodes can be colocated. Up to 16 x 128 or 2048 lines can be connected to this recommended maximum size single node.

7.6.3 VOICE SHARED MEMORY REQUIREMENTS

By employing four independent voice shared memory banks, 50 MBS of the total 100 MBS node traffic, (that is, the incoming traffic) can be split four ways. Each shared memory bank must then handle a worst case thruput of 62.5 Megabits per second (12.5 MBS input plus worst case output of 50 MBS). Assuming halfword access (16 bit wide) the word transfer rate to and from each bank must be nearly 4 MBS (memory cycle time should be no greater than 250 nanoseconds). Each bank must store about twenty frames of data. (Each bank must handle 4 T-1 full duplex trunks; however, due to the need to buffer approximately 2.5 times the data sent per frame to accommodate voice call movement within a dynamic frame and to provide buffering to accommodate clock differentials, 20 data frames capacity store must be provided). Based on a proposed 10 MS frame of 15,360 bits, approximately 20,000 words of memory (16 bit wide) is required per voice memory bank. The capacity is reasonable and, although, the speed requirement is somewhat faster than standard

memories of this capacity, it is not an unreasonable design requirement.

7.6.4 ICC REQUIREMENTS

Let us now examine the ICC unit and its modularity. The ICC performs more processing on data traversing its ports compared to the processing the BMX performs on voice traffic. The ICC design for the testbed employs state-of-the-art high speed memory chips and is a special design employing ROM control memory. The thruput achieved is 2 MBPS. Based on an estimated peak second data thruput requirement of nearly 12 MBS for a super node at least 6 ICC units must be accommodated. However, since each ICC data trunk must feed an integral number of BMX controllers, symmetry requires that at least eight ICC units service the maximum configuration of 16 BMX controllers. This works out very neatly to one ICC per processing bus. Since the estimated maximum number of data access lines to be accommodated ranges between 500-1000 each ICC should accommodate up to 128 access lines. This is reasonable and is in fact the line capacity of the ICC designed for the testbed. An inherent limitation of utilizing an ICC to service two BMX trunks is that each ICC trunk link can only sustain a continued trunk rate which does not exceed 250 KBS (allowing equal capacity for subscriber service) or about 16% of the total T1 trunk capacity (based on a 2 MBS total thruput). For most nodes, average data thruput ranges between 5-10% of the total voice thruput (based on 16 KBS voice rate), so this limitation is not an excessive constraint. However, some nodes may require an ICC per BMX if data thruput represents more than 10% of the total node thruput. This permits the average data rate on a trunk to reach 500 KBS, or 33% of the T1 capacity.

It is of key importance to explain how the ICC trunk rate can differ from the internodal trunk rate and why it is important to limit the ICC trunk rate. Although the internodal trunk operates at the T-1 rate, the data traffic can (and should) be limited to a lower rate. The BMX controller can buffer ICC data

and feed it to the trunk at the T-1 rate provided the average asynchronous data rate does not exceed the maximum ICC trunk rate. Normally the voice traffic occupies a substantial part of the trunk bandwidth, and the average asynchronous data rate will be less than the maximum ICC output rate. However, if there is little or no voice traffic on the trunk, the total trunk bandwidth would be available for asynchronous data. If the ICC puts data out at a maximum rate of 250 (or 500) KBS, it is then necessary for the BMX to fill the trunk with dummy idle characters to sustain the T1 trunk rate. Although this appears to be wasting bandwidth to fill the trunk, it is the recommended approach for the following reasons:

In general, average data thrupt at a node will run between 5 and 15% of the voice thrupt based on projected DCA traffic statistics (and assuming voice traffic rate is 16 KBS). Based on this, it is likely that at least 85% of the trunk will be normally filled with voice traffic. In this case, if no voice traffic is on the trunk, the bandwidth available for data is more than six times the required data bandwidth. If the data network is designed to handle this thrupt, the data network will be grossly oversized and very costly.

It is, therefore, proposed to determine ICC thrupt requirements on the basis of required data thrupt rather than full utilization of available trunk bandwidth.

This point is best illustrated by example. If a super node is being constructed with projected voice traffic of 1100 erlangs and an average voice rate of 16 KBS, approximately 16 T-1 trunks are required or a total thrupt capacity of nearly 100 MBS. If the same node must handle an average of 1300 packets/second, each packet 2000 bits in length and assuming

the traffic split is 33% tandem, 33% incoming and 33% outgoing, total average trunk thruput required is 3.5 MBS and peak second trunk thruput of about 8 MBS. For a 16 trunk node an ICC trunk data rate of 250 KBS is sufficient to handle the peak second data traffic.

If the voice data rate were to drop to 8 KBS, then only 8 T1 trunks would be required; however, the ICC rate for each trunk would have to be 500 KBS to accommodate the same data traffic. Based on these type trades, the following parameters are advocated for ICC design. The ICC should be designed to attain total thrupt capacity of 2 MBS which can be apportioned between ICC trunks and access lines in any of the following ways:

(1) 2 Trunks - [250 KBS max. trunk rate (full duplex)
+ 128 access lines Access lines total capacity 1 MBS]

or

(2) 1 Trunk - [500 KBS max. trunk rate (full duplex)
+ 128 access lines Access line total capacity 1 MBS]

or

(3) 1 Trunk only - [1 MBS max. trunk rate
No access line capacity]

or

(4) 0 Trunk - [Access line total capacity 2 MBS]
128 access lines only

7.6.5 SHARED DATA MEMORY SIZE AND SPEED

Since all traffic must pass thru the shared data memory, the shared data memory must handle up to 16 MBS of ICC traffic. It is advisable to provide sufficient processor memory access to process the peak total data thrupt. At least 4 MBS capacity should be allowed over and above the 16 MBS required for ICC access. A 16 bit wide shared memory with 750 ns cycle time provides over 21 MBS thrupt capacity. Use of two independent banks provide added thrupt margin (due to overlapping) and should be adequate.

A memory capacity sufficient to allow an average packet duration in node of 2 seconds (I/A packet duration will be much less and bulk packet duration may be greater) is a good rule of thumb. Throttling at data sources prevents memory overflow. Thus a total memory capacity of $2 \times 1300 \times 256$ or 665,000 bytes should be sufficient buffering for a super node. Three independent memory banks, each 256K bytes in capacity should be adequate shared memory for a super node. Since processors must also address private memory, a processor memory addressing capacity of Megabyte is recommended.

7.6.6 PROCESSOR CHARACTERISTICS

Processing of each packet will require a processor to perform between 400 and 2000 instructions. The number of instructions required is primarily dictated by two factors:

- (1) The power of individual instructions
- (2) The amount of processing required to convert a segment to a packet.

- The latter item is a critical factor. If the segment leader requires major code and format conversion to become a packet header, 300-600 additional instructions could be required to process the segment. Movement of text data for leader to packet conversion should be avoided (by proper choice of Leader/Header formats or by employing Front End Processing) because this would prove extremely time consuming. Segment disassembly and re-assembly, if required, will also significantly add to the packet processing time. Code conversion, when required to standardize header code of the packet is another significant processing load.

The first item, instruction power, is also important. Typical differences in basic instruction complement sophistication between a mini and a micro might range between 1.2 to 1.5 in favor of the mini. Typical instruction execution times range from 1.5 microseconds for a fast mini, to 4 microseconds for a microprocessor. Based on these general considerations and recognizing that other factors such as cycle stealing for data access to memory, and executive processing will cutdown the

overall thruput, it is reasonable to presume that a microprocessor can handle between 100-200 packets a second and a miniprocessor between 200 and 600 packets per second.

From these considerations and allowing for design margin and multiprocessing inefficiencies, a system containing from 8 to 16 microprocessors or 4 to 8 miniprocessors (depending on the power of the selected processor) should be adequate to handle the largest traffic loads (i.e., 1300 packets/sec). It is important to note that these estimates are based on maximum packet lengths of 2000 bits. Larger maximum packet sizes will reduce the processing burden, but at the expense of increased memory requirements.

7.6.6.1 MEMORY ADDRESSING CAPABILITY

Due to the relatively large amount of data buffering required, it will be necessary for each processor to access a large size memory. For the recommended configuration the large addressing capability refers to shared memory access. Memory addressing can employ base registers or other methods to extend capacity, however, the ability to develop a 20 bit address is essential. Several miniprocessors, but no microprocessors currently available, can directly address this size memory. It would, however, be feasible to introduce a hardware controller interfacing the shared memory, and the processor private memory. This controller would provide the necessary control to perform DMA-DMA transfers between processor private memory and any area of the shared memory. Using such a controller effectively extends the data access capability of the microprocessor, and is a rapid and efficient way to transfer the data access capability of the microprocessor, and is a rapid and efficient way to transfer the data. The processor private memory would contain only the most frequently used processing programs, and would buffer that part of the single packet being processed by that microprocessor. Many current microprocessors can address 65 Kilobytes of memory which is an adequate capacity for the private memory if used mostly for control only. The need

for a memory extender device such as this type of controller just discussed is an important factor when comparing the cost of a system developed with microprocessors versus use of a mini-computer which may already incorporate the required memory addressing capability.

7.6.6.2 DMA CAPABILITY

The processor should provide at least one DMA access port to memory.

7.6.6.3 ASYNCHRONOUS MEMORY OPERATION

It is probably desirable to select a processor whose instruction speed can adapt to the memory cycle time. This permits upgrading of processing power as memory speed improvements are realized, and has the particular advantage of permitting use of extra high speed control memory for time critical portions of the packet processing program. This capability is common in present day processors and coincides with our objectives of developing a flexible system growth capability.

7.6.6.4 EXTERNAL INTERRUPTS

As will be discussed in the software architecture a means will be provided to consolidate external interrupts. The number of required external interrupts can be as low as one, however, good design practice suggests that the selected processor be able to accommodate from 4-6 independent external interrupts.

a) Interrupt Enabling

The processor should have the capability of enabling or disabling, by software means, any of its external interrupts.

b) Interrupt Speed

If interrupts were to occur on a per packet basis and a processor handles say 400 packets per second, the average interrupt period is 2.5 millisecond. An interrupt speed of 20 microseconds or less is reasonably

achievable and presents less than 1% overhead load on the processor.

c) Multilevel Interrupt Structure

A desirable, but not absolutely essential processor feature is some interrupt on interrupt capability wherein certain higher classes of interrupt can interrupt ongoing lower level of interrupt processing. However, if packet switch and circuit switch or message switch interrupt processing functions are not performed by independent processors (independent processors is the recommended approach), then a multilevel interrupt structure becomes a necessary processor feature.

7.6.6.5 SOFTWARE SUPPORT FEATURES

- . High Level Language - It is highly desirable to select a processor which offers an efficient compiler which permits the use of a high level language for coding.

The software cost will be an important percentage of the total cost even if amortized over 60-70 nodes. Use of a high level language is highly desirable. It is noted that this capability is being offered by at least one microprocessor manufacturer, so the desirability of this feature does not preclude employing microprocessors.

- . ~~Assembler~~ - Certainly the more critical areas of the software code will have to be programmed in assembly language. An assembler written in a commonly used language (e.g., Fortran) is a necessary support package. A self-assembler feature is desirable but not a firm requirement.
- . Simulation - A strong support package is required to permit rapid software test and checkout. Processor simulation is a useful tool wherein internal processor

events can be monitored employing a larger processor with special monitor capability which simulates the purchased processor. Other areas of software support in the areas of development and debug include programs for:

- Mass dump
- Mass store dump
- Program patching
- Program trace
- Program update
- Table Interpreters

- . Diagnostics - A good self-test package which locates processor failures is a highly desirable processor feature.

The emphasis of the instruction complement is on data handling and the need for powerful instructions to support the following operations:

- . Data Moves to and from memory
- . Data Exchanges between private and shared memory
- . Logical Instructions with branches conditional and unconditional
- . List Searches
- . Bit and Byte manipulations
- . Incrementing and decrementing tally counters
- . External hardware control

8.0 FUTURE NODE ARCHITECTURE - SOFTWARE

8.1 GENERAL

This section discusses the design of the software architecture to support the future universal node. The software must be designed based on the recommended hardware architecture.

(As illustrated in Figure 7.20). The universal node will provide digital voice switching and data

packet switching. The digital voice switching must provide for subscriber access and trunks. For the trunks, all inter-nodal signalling will be via data packets. However, access subscribers will be accommodated using the user equipment signalling techniques. Thus, for the subscribers, line scanning (to detect on/off hook) and registers (to detect/collect dial digits) must be provided. As in traditional circuit switches, this equipment is pooled and assigned as required by the control processor. The equipment inputs signals to the control processor via DMA or interrupts. Since the access load is limited, the hardware architecture recommended (depicted in Figure 7-20) indicates that the common equipment for the circuit switch access function need not interface with all processors in the node. Thus, the recommended architecture can employ a single processor (with a back-up) for control of the circuit switching function and a multiprocessor configuration for the packet switching function. In keeping with the hardware architecture, the software is partitioned as follows:

- (1) A circuit switch processor will perform the processing for the access signalling, BMX channel assignment and BMX interface.
- (2) N packet switching processors will perform the processing for segment and packet protocols, routing and traffic flow control.

The thrust of this section is the evolution of the software architecture for the N packet switching processors. The packet traffic requirements are such that a modular building block technique is recommended. Thus, the software must be designed to run on "N" processors and handle "L" lines with a throughput of "P" packets per second. Based on the actual number of lines and packet throughput, the number of processors for a given node will be determined. However, the software structure must be such that same software package can be used on each of the nodes. The following paragraphs will examine the executive structure, the interrupt structure and the processing allocation.

8.2 EXECUTIVE STRUCTURE

For background information, three basic types of executive structures for a multiprocessor are discussed. The structure recommended for the packet switching function is then presented.

8.2.1 MASTER-SLAVE ORGANIZATION

In the master-slave operating mode, the executive program is run on one processor which is the master. The master performs executive, supervisory and control operations for the system. It delegates tasks to the slave processors which are equally able to perform all tasks. Failure of a single slave processor has minimal effect on the system providing that the task which was in progress on the failed processor can be restarted. Failure of the master processor faults the entire system and so provision must be made for restart with one of the slave processors as the new master.

The master-slave mode is probably the easiest to implement in terms of both software and hardware requirements. The main disadvantage is that slaves may frequently be forced to sit idle until the master gets around to assign the next task. The master load can become very great so that it will lag the slaves, causing inefficiency. One approach to minimize this problem is that of asymmetric hardware whereby the master processor is larger and more powerful than the slaves.

8.2.2 AUTONOMOUS ORGANIZATION WITH MULTIPLE EXECUTIVE

In this type of system each processor is autonomous; there is no overall controller or master. Each processor has a preordained task to perform and it runs its own executive programs which supervise and control execution of the task. Failure of a processor can result in loss of a vital function which, therefore, must be assumed by one of the surviving processors. Such transfer may be difficult without a master processor.

Executive programs can reside in shared memory or in each processor's private memory. If stored in shared memory, contention is increased and reentrant code must be used because numerous processors will execute the same programs. If stored in private memory total memory requirements increase. The best choice is probably an optimum combination of shared and private storage of the executive.

An efficiency loss analogous to that of the master-slave system can occur in this system. Since task assignments are semi-permanent, it is possible for some processors to be overloaded while other are idle. Dynamic task allocation may be considered but without a master processor in control would be difficult.

8.2.3 ORGANIZATION WITH FLOATING EXECUTIVE

In this concept, all processors are equally autonomous. There is no single master processor nor are the individual processors assigned fixed tasks. Each processor executes supervisor functions connected with the task it is presently executing and those required to get a new task when the present task is finished. Any processor can perform any of the tasks. Overall system control floats from processor to processor. Only one is in overall control at one time although more than one may be performing supervisor functions related to its current task.

This concept offers the most efficient use of assets but is undoubtedly the most difficult to implement. It provides high availability and graceful degradation because all processors are equal and autonomous. The floating executive idea makes for a difficult programming job whether one considers the coding, debug or software maintenance aspects.

8.2.4 EXECUTIVE STRUCTURE FOR PACKET SWITCH

When two or more processors are to work simultaneously on a job, the design must partition jobs so that parallel processing can be

utilized effectively and also accommodate serial performance of functions when necessary. That is, the system must prevent a processor from starting if it depends on another process which has not yet been completed.

For example, all lines in the switch are idle and then on one line two short packets are received. The design must not allow each packet to be processed by a separate processor simultaneously. Time wise, the second packet must be processed after the first, since its validity may depend on information in the first packet. Thus, it is not important which processor actually processes the second packet as long as it is processed after the first.

Since items from the same line usually must be handled serially, a partition by lines forces serial processing per line but allows parallel processing for independent lines. This kind of system partitioning is of the generic type discussed in 8.2.2 where the preordained task is to service "N" specific lines. To not use this system in favor of the more complex structure of a floating executive there must be some very strong reasons. A loss of 5 to 8% in efficiency is not sufficient reason to embark on a much more complex design structure.

In order to modify the pre-assigned tasks, due to processor failures, a quasi-executive function must be included in the system. This function can be assigned to anyone of the processors. It is only exercised upon processor failure or return to service. At any time, only one processor is assigned the system executive function. This processor is responsible for controlling the partitioning of the system. For example, if "L" lines are handled by "N" processors, then normally each processor serves L/N lines. If the processor fails, then the L/N lines associated with the processor must be handled by the other processors. This involves changing the partitioning so that each processor serves $L/(N-1)$ lines.

8.3.0 INTERRUPT STRUCTURE

Over the past two decades, computer interrupt structures have evolved from none (program had to periodically sample) to one level (some I/O needs help) to multi-level (each external interrupt can cause its own unique interrupt). The interrupt capability allows the timely deployment of processing power to the proper sub-task. Part of the system/software architecture is the assignment of priority levels to the various external interrupts, and the decision as to how much processing should be done at each level and how much deferred to a lower level.

This section will discuss basic methods of handling interrupts and then present the recommended software approach. Basically, there are four ways to handle interrupts:

- (1) No interrupt capability
- (2) All processors handle interrupts
- (3) 1 dedicated processor handles them
- (4) 1 processor (ring around)

8.3.1 NO PROGRAM INTERRUPTS

A multi-processor system with no hardware interrupts would be a return to the primitive method used in the first computer systems. The program would have to be designed to sample for the occurrence of external events at a rate consistent with the duty-cycle of the external event. Thus, the whole program must be structured so that the required sample rate is achieved. This would appear to place an undue burden on the software design and maintenance. That is, the running time of every program segment is critical to the system's ability to service higher priority demands. Also, since all processors can do any task, more than 1 might be servicing a task (normal interrupt type) that requires read/write access to the same information. Thus, some interlock scheme must be developed and deployed.

8.3.2 PROGRAM INTERRUPT TO ALL PROCESSORS

On the other end of the spectrum we could allow the device requiring service to cause an interrupt in all system processors. Since we would want the demand to be serviced only once, the processors would then have to coordinate among themselves who would handle the interrupt. Thus, it would result in one processor servicing the interrupt while all processors would be involved in the processor selection process. This overhead would be wasteful.

8.3.3 PROGRAM INTERRUPT TO ONLY ONE PROCESSOR

This would seem to be the most efficient method. There exists two possibilities; one, a dedicated processor handles all interrupts or two, the responsibility to handle the interrupt is rotated around the processors.

This method removes the critical timing problem from the program segments (no interrupts) and reduces wasted processor overhead (all processors get interrupt). Of course, the system must have some protection scheme to insure that the interrupt is serviced (silent death protection).

If a dedicated processor handles all interrupts, then there must be means for assigning that job to any of the physical processors. The hardware interrupt mechanics must be capable of sending interrupt to any of the processors. Thus, there must be some selection logic which steers the interrupt to the appropriate processor. This would probably be the same element for a dedicated processor or rotated service (since the dedicated must be any processor). The difference is in the use. With rotated service, the assigned processor is changed every interrupt. With dedicated, the assigned processor is only changed for failure/recovery.

With rotated service, protection must be provided for interrupt interference - that is a second interrupt occurring (for the next

processor) before the processor is finished with the current interrupt. One method of avoiding this is with an interrupt stack for the I/O demands and a separate time dependent interrupt source, which indicates check stack.

8.3.4 INTERRUPT STRUCTURE FOR PACKET SWITCH

To conform to the executive structure of Section 8.2 it would be desirable to have each processor handle the interrupts for the lines it services. However, since the lines that a given processor can handle are variable and the modularity of the front-end device is independent of the number of lines assigned to any particular processor, it appears advantageous to use one processor to steer the interrupts to the appropriate processor. In this manner, all hardware interrupts can be switched as a unit to one processor. This processor will then generate program controlled interrupts in the proper processors.

The hardware communication device(s) places interrupt items in a common stack in shared memory. In addition, the hardware causes an interrupt. This interrupt is switchable to any of the "N" processors but at any given time is assigned to only one. The interrupt servicing processor takes the item(s) from the common stack and places it in the unique stack for the processor that handles the line which causes the interrupt. It then causes an interrupt in each of the processors which received a new item. In this manner, each processor handles the interrupts for the lines it services.

8.4 PROCESSING ALLOCATION

As discussed in the hardware architecture section, the first division of the processing function is between voice and data. The processing function identified as voice handles the access signalling with local subscribers, the trunk signalling via data packets and interfaces with the BMX type device(s) to coordinate the inter-node transmission frame. The processing function

identified as data handles all packets and segments. As indicated, the voice processing function is envisioned as being performed by one processor with a back-up processor; and the packet switching function is envisioned as being performed by multiprocessors.

8.4.1 PACKET SWITCHING

As discussed in Sections 8.2 and 8.3, the packet switching function is performed by a multi-processor system with each processor handling I/N of the lines. In addition, the interrupt items are steered to the processor which handles the interrupting line. Within each of these processors the software is organized in a three level priority structure. These are called; interrupt service, interrupt processing and base levels. The interrupt service level (ISL) is called whenever a new item is added to the processor interrupt stack by the steering logic. This level performs a minimum amount of processing, (example - give input line new Buffer Block) and queues the packet/segment received or transmitted for processing by the interrupt processing level. The interrupt processing level (IPL) is the lower priority interrupt level. It performs the actual processing and routing of new input packets/segments. The reason for breaking the interrupt processing into the levels is to allow rapid response for time critical operations. The interrupt service level (ISL) can interrupt IPL. The base level is the normal processing level of the processor and the programs in this level run whenever there is no interrupt level active. The base level processing scans all lines and initiates all non-interrupt driven processes.

8.4.1.1 INTERRUPT SERVICING LEVEL

The ISL programs are called upon to service all interrupts generated by the lines assigned to this processor. The two major interrupts are for: (1) an output block transmitted and (2) an input block received. For an output interrupt, ISL will

queue the block for processing by IPL. For an input interrupt, ISL will queue the received block for processing by IPL and assign a new buffer block to the input line.

Since the pool of blocks is shared by all processors, a safeguard must be utilized to insure that the same block is not assigned simultaneously by two processors. A common vacancy chain is used and only one processor is allowed to be using the chain at any time. This is accomplished by a busy or lock-out flag for the chain. When any processor needs a block, it first checks the busy flag, and if on, it waits. If the chain is not busy, the processor sets the busy flag, gets a block and then unbusies. The actual time it takes to get a block from the chain is very short, (less than 10 msec.) thus, no processor must wait long to gain access.

8.4.1.2 INTERRUPT PROCESSING LEVEL

IPL programs are called upon to process all blocks which were transmitted or received by the switch. ISL services the interrupt item and queues the referenced block for processing by IPL. IPL consists of the two main programs, process transmitted block and process received block. (Block can be either a packet or a segment).

8.4.1.2.1 PROCESS TRANSMITTED BLOCK

This program processes blocks which have been transmitted by the switch. For each block, it determines the output line utilized and then places the block on a wait acknowledgement Q for the line. The Q contains (for the line) all blocks transmitted but not yet acknowledged by the receiver. In addition, this program checks the transmission back-log Q's for the line and selects the next block for transmission.

8.4.1.2.2 PROCESS RECEIVED BLOCK

This program processes the input blocks which have been received and queued by ISL. The blocks could have been received

on either an access line or a data trunk; and could be of two types: Data blocks or link control blocks.

(a) TRUNKS

The trunks at the switch receive packets which are either 1) data packets destined to a local host, 2) data packets destined to another node, 3) signal packets for the inter-node trunk, or 4) line control packets for the data channel (which contain no data).

Data Packets

Line control processing is performed for every data packet received. The received packet's control field (NS and NR) is processed, and if valid, the packet is accepted. If the packet is not accepted, it is discarded. (Buffer released back to available pool.)

If the packet is accepted, it is then routed to the appropriate output line. The routing procedure used is based on the destination node specified in the packet header. If the destination node is not this node, then the packet is routed via the trunk routing table. The destination node number is used as an index to address the proper routing entry in the table. The entry indicates the trunk to be used to reach the node. The old link control fields are cleared and the packet is placed on the transmission Backlong Q for the data trunk.

If the destination node is this node, then the packet is routed via the destination line number specified in the packet header to the appropriate access line. The packet is placed on the Routed Packet Q for the access line.

Signal Packets

These blocks have no data to be switched and are used to perform the link control procedure. This program initiates the required protocol response and releases the block.

Signal Packets

The signal packets are transmitted between nodes as data packets. They receive the same link control processing and are not recognized as signal packets until the packet is accepted. The header of the packet marks the packet as a signal packet for the inter-node trunk which serves the data channel. The packet is placed in a queue to be processed by the voice processor.

(b) ACCESS LINES

The access lines at the switch receive segments which are destined to another host. These lines also use control procedures. The segments are the information field of a data block. The received segment's control field (NS and NR) is processed and if valid, the segment is accepted (if not accepted, the block is released.) If the segment is accepted, it is queued to the access line that serves the source host until the segment protocol allows it to be transmitted to the destination.

Access lines can also receive link control information which is utilized by the input program to initiate the required protocol response.

8.4.1.3 BASE PROCESSING LEVEL

The base level program scans the communication lines; processes each access lines Queue of packets and segments, performs output processing for all lines, and monitors timers and operator commands for all lines.

8.4.1.3.1 ACCESS LINES

(a) Routed Packets

New packets queued to this line are either data for one of the logical connections in progress or an attempt to establish a new connection or a control packet. The new packets which are data for an established connection will be reassembled into a segment and the appropriate segment protocol initiated. The segment is then placed on the transmission Backlog Q.

The new packets which are an attempt to establish a new connection will cause a connection to be established if resources are available. The packet will be re-assembled into a segment and a connection confirmed control packet will be sent back to the source. The new segment is placed on the back log queue.

The new packets which are segment protocol control packets are processed accordingly and a control segment is placed in the transmission backlog Q if necessary.

(b) Received Segments

New segments received on this access line are either data for one of the logical connections in progress or an attempt to establish a new connection.

The segments that are for already established connections are linked to the corresponding connection entry. Based on the sequence numbers in the connection entry, the segment either can be forwarded to the destination node or must await for a control packet. The processing for segment forwarding depends on whether the destination host is local or

remote. The segment is packetized and if the destination host is local, the packet is placed on the routed packet Q for the destination host. If the destination is remote, the packet is placed on the transmission backlog queue for the trunk to the remote node.

8.4.1.3.1 OUTPUT PROCESSING

The output processing consists of transmitting blocks from a lines Backlog Queue. In performing this function, several ADCCP functions must also be performed. (Data blocks are each assigned a transmit sequence number (NS). If the unacked maximum for a line is reached, transmission on that line pauses until blocks previously transmitted are acknowledged.

8.4.1.3.3 MONITOR

Base level programs also perform the time-out functions for both the ADCCP and Segment Protocol. ADCCP timers are checked every Scan while Segment Protocol timers (logical channel timers) are checked only when an overall time has expired.

The Node operator can perform several control functions on each line (Ex: put line in service, out of service, set procedure and security level etc.). The execution of some of these functions requires phase in with the activity on the line. These functions are coordinated by the base level programs.

8.4.2 VOICE SWITCHING

The voice switching function is performed by one processor with one back-up processor. The voice processing function is comprised of access line processing and trunk processing.

8.4.2.1 ACCESS LINE PROCESSING

The processing for the access lines is the traditional circuit switching type processing; ON/OFF Hook Processing, Seize/Release Codeword Election, tone sending and digit detection. These programs schedule and scan each access line, assign register/sender equipment as required and accumulate the dialed digits. If the call is to be extended via a trunk to another node, a signalling packet is prepared and Q'ed for the trunk processing program.

8.4.2.2 TRUNK PROCESSING

The trunk processing consists of signalling with the distant node to control the establishment and disconnect of voice calls over trunks which utilize dynamic channel allocation. This processing includes exchange of signalling packets via the data bandwidth, the allocation of voice bandwidth and the control of the BMX type device.

For a locally originated call, the program determines if voice bandwidth is available on the primary or alternate paths. If so, a signalling packet is Q'ed to the data channel for that inter-nodal trunk. This packet will be transmitted via the packet switch. Thus, the signalling is protected by the normal protocol used on the data channel.

After the signalling is completed, the call is added or deleted to the frame by changing the frame list which the trunk utilizes. A corresponding change is made to the bandwidth of the data channel in order to keep the frame size constant.

8.4.2.3 CONTROL OF BMX

The BMX is a Buffer Matrix which is to be used to accomplish dynamic channel allocation. The BMX functions are controlled and coordinated by the Processor software. The buffers for the voice and data are assigned by the switch software. In addition, the switch software prepares a frame transmission list for each trunk. On output, the BMX uses the frame list and the buffers associated with each entry in the list, to produce a synchronous frame of bits on the trunk. The frame length is determined by the sum of the byte counts for each entry in the frame list. On input, the BMX does the inverse. It takes the bit stream and fills buffers in accordance with its receive frame list. Again

AD-A049 619

RCA CORP CAMDEN N J

F/G 17/2

STUDIES FOR DEVELOPMENT OF A UNIFIED NODE EMPLOYING DYNAMIC CHA--ETC(U)

DEC 77 B PATRUSKY, K BODZIOCH, R CHAN

F30602-75-C-0109

UNCLASSIFIED

RADC-TR-77-380

NL

5 OF 5
AD
A049619



END

DATE

FILMED

3 - 78

DDC

1
this frame list was prepared by the switch software.

When a new voice call is established, the switch software includes the new voice call in the frame list for the trunk and reduces the byte count for the data entry. When a voice call is disconnected, the switch software removes the voice call from the frame list for the trunk, increases the byte count for the data entry. Frame size is controlled by the sum of the byte counts of all entries in the frame list. The frame length is kept constant by having the byte count allocated to data equal to $F - V$, where F is the total number of bytes in a frame, and V is the number of bytes assigned to voice.

Considering that the T1 trunk may at some points use satellite links to go from one node to another, adding or deleting one voice call per frame change may be too slow. Thus, the scheme used will establish one node on the line as the coordinator. This node will periodically initiate a frame change for all calls requiring an addition or deletion. The new frame structure will be indicated by a signalling packet and then the frame change will be accomplished via CCIS codes.

9.0 RECOMMENDATIONS FOR FUTURE STUDIES/DEVELOPMENT

The following represents a shopping list of testbed enhancements or studies necessary to further advance the development of the unified node. The first recommendations results from the stated intent of the government (RADC) to move the testbed and takeover testbed operation.

Task 9.1 REPLACEMENT OF LAB PROCESSOR

An effort is being planned to incorporate a standard off the shelf minicomputer to replace the RCA lab processor in the testbed, since it is hoped that cost to the government would be less to replace this processor than for the government to obtain and maintain this lab processor which is a "one of a kind" processor. The problem of software capture and unique hardware interfaces complicates this task.

Task 9.2 ENHANCEMENT OF HOSTS

Enhancement of hosts to incorporate Terminal Interfaces.

The testbed host equipments developed in this phase of the program serve terminals and provide for segment interchange to the developed node. However, the interfacing terminals (VDU and cassettes) must input and receive individual segments. It is probably desirable in order to use the testbed in an operational environment that the host design be enhanced to permit the operator(s) to enter messages (leader followed by message text) rather than individual segments. Enhancement involves adding additional RAM memory to the host and development of additional host software.

Task 9.3 ADDITION OF A COMSEC SUBSYSTEM

Would permit handling secure phones (however would introduce problems of clearances, physical security, etc.).

Task 9.4 Addition of gateways from the testbed to existing common user networks specifically

ARPA

AUTODIN (Partially in place)

Task 9.5 SATELLITE INTERFACE

The problem of trunking via satellite has not been studied in this phase of the program. It is likely that special interfaces will be required. It would be possible to use the testbed configured as a subnetwork which includes a leased satellite loop to develop and test such an interface.

Task 9.6 Requirements studies leading to developing certain areas of the unified node specification not investigated in this phase of the program namely:

Tech Control Requirements

Node Management Requirements

Network Control Center Development Requirements

COMSEC Requirements

Gateway Requirements

Task 9.7 FURTHER EXPERIMENTAL TESTS

The simulator developed in this phase of the program can yield a great deal of data associated with the effect of dynamic channel allocation on different node configurations. Much of the inherent capability of this developed tool has been untapped. Delay and Thruput statistics can be developed for different node sizes, line and trunk rates, line and trunk quantities, node buffer capacitors, node processing capability, throttling thresholds, etc.

Task 9.8 ROUTING ALGORITHM

Via use of the experimental testbed, a voice routing algorithm particularly suitable for trunks employing dynamic channel allocation could be developed. Trunk bandwidth occupancies, data backlogs and number of hops would be factors influencing the algorithm.

Task 9.9 ADDITION OF TERMINALS TO TESTBED

Enhancement of the testbed to accommodate many more terminals.

Task 9.10 NETWORK STUDIES

Network studies employing unified nodes. Examination of the air force base communication application and DCS backbone and access area requirements for the purpose of configuring the unified nodes into network configurations.

Task 9.11 TEST SUBNETWORK

Development of a small test subnetwork employing three unified nodes connected via leased T-1 trunks.

Task 9.12 STAND-ALONE BMX

Modification of the BMX to operate as a stand-alone multiplexer employing dynamic channel allocation.

Task 9.13 MULTIPROCESSOR CONFIGURATION

Enhancement of the testbed by addition of processors and shared memory leading to developing the recommended node architecture.

Task 9.14 SYNC/FRAME CHANGE PROCEDURE

Incorporation and testing of the recommended sync/frame change procedure to validate the recommended approach. Bit integrity tests on voice calls would be included in this group of tests.

9.15 VOICE/DATA PRIORITIES

A study of various rules involved in arriving at an algorithm for a programmable dynamic maximum voice limit. This dynamic property is desirable in a situation where data traffic is very heavy. To prevent too large a backlog of data transmission, the maximum voice limit may be contracted by an amount based upon certain rules (to be determined) to allow transmission of some data at the expense of any new incoming voice calls. It should be noted also that in the reverse situation where traffic of voice calls is heavy and data calls light, the voice limit may be expanded to allow more voice calls.

10.0 CONCLUSIONS

The conclusions are discussed in "Summary of Key Findings" which follows the introduction of this report.

GLOSSARY

AA	Accountability Acknowledgement - See Autodin II Spec. - Segment Protocol Term
ADP/T	Automatic Data Processing/Telecommunications
ADCCP	Advanced Data Communications Control Procedures. The link protocol used on all links in the ADPT System. Node to Node and Host to Node.
Access Line	A physical line from a node to a network user - e.g. to a Host.
ALLC	See description of Connection Entry Table
ACK	Acknowledgement - ADCCP Protocol Term. Notifies a sender on a link that a block has been received.
ADCCP Address	A one byte address assigned to a user of a unique link. See ADCCP Protocol. This is not a network address.
ATB	All Trunks Busy. Voice Signalling Control indicating that a trunk was not available for connection of a voice call. Awaiting Signalling Queue. See BMX Nodal Line Table.
BASE Program	The lowest interrupt level program - runs only when no interrupts are in progress. In the Data Switch the Base Program is everything except the BMX and ICC interrupt programs.

BMX Nodal

Line Table See table definitions in handbook.

BMX Buffer Matrix

Backlog Queues Queues of data blocks waiting output service.

BMIP BMX interrupt program (part of ADPEXC)

BALE Buffer address list entry - see ICC spec.

BIT Binary digit

BYTE Eight contiguous binary digits

BMX Physical

Line Table See table definitions in handbook.

Block by Block A level of end-to-end segment throttling when blocks are asked for by destination node one at a time.

BULK Segment protocol term. Type of data traffic - large amount of slow moving data.

BSQ BMX Signalling Queue - See BMX Nodal Line Table.

BUFFER A 256 byte storage area in 12100 memory. Assignment of buffers is regulated using the Queue Table.

BLOCK Depending on context can mean - packet segment or buffer.

Buffer Block Number	A number designating a buffer area.
Connection Entry	See table definitions
CHS	Channel Scan Program
Channel Scan Program	That program in the software that scans lines looking for activity.
CSIP	Circuit Switch Interface Program. The program that interfaces with the CSU to process call connects and disconnects.
C/S	Circuit Switch
CSU	Circuit Switch Unit (software program).
CC	Communications Channel
CCIS	Common Channel Interoffice Signalling
CC	Connection Confirm
CRC	Cyclic Redundancy Check - See ICC Spec.
Control Segment	A data block used by Segment Protocol to control data flow.
Continuous	A throttle level for Segment Protocol which allows both destination and originating nodes to have two blocks in progress.

Connection per Block	
Control Memory	Portion of BMX memory - See BMX spec.
CI	Call Initiate
CU	Called unavailable (e.g. busy).
CNF	Change next frame - indication in CCIS portion of BMX frame indicating frame change.
Capture	To reuse existing software in a new program.
Connect	To establish a connection between two voice users.
CLG	Calling - the user dialing the phone.
CLD	Called - the user being called.
Closed Subroutine	A software subroutine that can be called other software and then returns to that software.
COS	Channel Out of Service
Dynamic Channel Entry	See Table definitions.
Data Buffer	See Table definitions.
Data Switch	The ADPT Software package that performs packet switching.
DS	Data Switch

DCA	Dynamic Channel Allocation
Data Switch	
Control	See software spec. in design handbook.
Data Memory	Part of the BMX memory. See BMX Spec.
Dummy Voice	Unused voice channels used to fill-up the BMX frame.
DIS	Disconnect
DIS Reply	Signalling Term - A reply to a disconnect request indicating successful disconnection.
DIS Repeat	Signalling Term - A reply to a disconnect request indicating that the request should be sent again.
Delta-Mod	Delta-Modulation. A method of converting analog voice to a digital bit stream.
Disconnect	To break a connection between two voice users.
Data Subscriber	A user of the BMX that passes data traffic instead of voice traffic. In this case the ICC Dummy Voice. A voice called established on a BMX trunk to fill unused bandwidth.
Executive	The control program of the software which schedules I/O tasks to be performed and initiates base level programs.
EOM	End of Message. Segment Protocol Term. A segment that is the last segment in a message.

EOM/ETX	A special EOM that indicates that no more messages are to be read from tape.
EOMSNT	See Connection Entry Table - EOM SENT
EOMETX	See Connection Entry Table
EOMR	See Connection Entry Table - EOM Received
Final	ADCCP Protocol Term - See ADCCP spec.
F	Final
FCC	Frame Change Complete. BMX processing term. See BMX Nodal Line Table
FCIP	Frame Change In Progress. See BMX Nodal Line Table
Host	A network user capable of sending and receiving data traffic.
Header	The leader of a data packet plus fields used by the network to switch and control the packet.
IDA	In delivery acknowledgement
ICC	Interactive Communication Channel
ICMS	Integrated Circuit and Message Switch
ICC Nodal Line Table	See table definitions.

ICC Physical	
Line Table	See table definitions
Interrupt	To stop the BPU's processing and force it it to service an external event (e.g. termination of an I/O).
Leader	That portion of a data block provided by the access user in order for the system to process the block.
Logical Channel	The logical link between two access users to the data network.
List	A list of those items which make up a BMX frame.
Nodal Line Table	See Table definitions.
New Segment Queue	See Table definition - ICC Nodal Line Table.
N(S)	Sending ADCCP Sequence number of a data block.
N(R)	Expecting ADCCP Sequence number of a data block.
NC	No change
NF	New frame
Nodal Work Area	See Table definitions
OOB	Out of Band Signalling

Pseudo Host	The software module that interfaces Autodin terminals to the packet switch via the message switch.
Poll	See ADCCP Protocol specification
P	Poll
Packet	A block of data framed by flags on an ADCCP link.
PROC	PROC Bit in software indicates ICC is processing blocks for a line.
PV	Pseudo Voice
Pseudo Voice	The voice connection established to support a tandem voice call at a node.
Queue Table	See Table definitions.
RFNM	Request for next message.
Retransmission	To retransmitt a packet due to an error detected by the ADCCP Protocol.